SCHEDULING PROBLEMS FOR FRACTIONAL AIRLINES

A Thesis
Presented to
The Academic Faculty

by

Fei Qian

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
H. Milton Stewart School of Industrial and Systems Engineering

Georgia Institute of Technology
December 2010

SCHEDULING PROBLEMS FOR FRACTIONAL AIRLINES

Approved by:

Dr. Ozlem Ergun, Advisor
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Ellis Johnson, Advisor
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Joel Sokol
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Pinar Keskinocak
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. John-Paul Clarke
Daniel Guggenheim School of Aerospace Engineering
Georgia Institute of Technology

Date Approved: Dec 15, 2010

ACKNOWLEDGEMENTS

I am grateful to my advisors, Dr. Ozlem Ergun and Dr. Ellis Johnson, for their encouragement, guidance and support. I also would like to thank my committee members for their help and support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Daily operations of fractional airlines involve scheduling airplanes and pilots to satisfy customer demand and minimize the operating cost. In this thesis, a column generation based approach is proposed to solve this problem efficiently and return near optimal schedules.

Crew tours are building blocks of our approach, and our approach is focused on exploring more feasible tours than other approaches. In particular, all elements of a crew tour are optimized during the preparation and tour generation procedures, including duties, arcs, crew travelling to airplanes, crew picking up airplanes, and crew's travels to home bases etc. Moreover, time windows of customer-requested flights are handled exactly, and generalized to time window and crew time window of duties and tours. Furthermore, time windows of tours are contained in the MIP formulation to ensure more feasible connections between tours.

In the pricing subproblem, an efficient constrained shortest path algorithm is proposed, which is necessary for our model and also provides extensibility for incorporating more complex constraints in the future. Computational results of our model show very small optimality gaps and consistent improvements over the model used in practice. Moreover, restricted versions of our model that have fast running time are provided, thus very desired in the case that running time has more priority than solution quality.

In order to understand the demand, data mining of demand data is presented and analyzed. Moreover, a recovery model is proposed to deal with unscheduled

maintenance in practice, by reserving airplanes and crews in the model. Computational experiments show the advantage of the recovery model, in the case of simulated unscheduled maintenance and comparing to models without recovery considerations.

CHAPTER I

INTRODUCTION

1.1 Problem Introduction and Background

The first fractional aircraft ownership program was introduced by Executive Jet Aviation Corporation in 1986, and it has grown steadily since then. With a fractional aircraft ownership program, owners can buy as little as 1/32 shares of an aircraft and have the benefits of private aviation. The percentage of an owner's share is translated to the maximum number of hours that this owner can request each year. For example, a full ownership equals to 800 hours of flight time, and a 1/32 shares equals to 25 hours of flight time. The contract for a share is typically 5 or 6 years, after which the owner can sell the share back to the management company or choose other options.

The management company manages and maintains aircrafts and hires pilots. Owners contact the management company in advance, as little as a few hours before the departure, to request flights that are virtually from anywhere to anywhere (mostly domestic) and depart at anytime. Owners must pay the management company the cost for the share, monthly management fees, and hourly cost for the flying time of the requested flight. The hourly cost may also include variable fuel surcharges. Note that the management company may also provide other types of on-demand services. We will refer to this type of fractional ownership program management companies as fractional airlines in contrast to commercial airlines and charter airlines, and we will denote airplanes managed by fractional airlines as fractional airplanes.

Fractional airlines have unique characteristics and target a distinct group of customers. In the following, we will show some advantages of flying with fractional airlines comparing with all other ways of flying.

- Commercial flights. Fractional owners can save total travel time, avoid jammed commercial airports, and enjoy the service and convenience of flying on a private airplane.

- Charter flights. Fractional owners have the benefits of partially owning an airplane (tax purpose etc.), can fly on their preferred type of airplanes, and can have the guaranteed quality service. In addition, since booking a charter airplane is basically "renting" an airplane and crew, there may be extra charges. For example, waiting time is charged by charter airlines, while fractional owners are not charged for any waiting time. If owners fly frequently, the average flying cost with fractional airlines is lower.

- Owning an airplane privately. Fractional owners can save the cost related to hangar, crew, maintenance etc., especially if they don't fly very frequently. And more importantly, a fractional owner can have multiple trips departing at the same time, which is impossible when owning a single airplane privately.

- Air taxi service. Fractional owners don't need to share the cabin with other customers, and they are always served with direct flights (except for necessary fuel stops). Moreover, a fractional owner's requests will never be rejected by the management company. But this may happen when customers request flights from an air taxi company. Also airplanes used in fractional airlines are usually larger and can fly over a longer distance.

The crucial point in the planning for fractional airlines is that owners are charged by their flying time, besides the fixed charges that are the same no matter owners fly or not. Therefore, the management company has to absorb additional operating cost such as dead-heading (flying without passengers on board) and possible charter flights. These additional costs are inevitable: owners request flights that are independent of the availability of the resources (airplanes and crews); one fractional airplane

may be owned by multiple owners, and owners of the same airplane request flights independently. For example, assume that a fractional airline has only a single airplane and two owners. If one owner requests a flight from airport A to B departing at 7:00, and the other owner requests a flight from airport C to A departing at 7:30, then it is impossible for the company to schedule its only airplane to satisfy both demands. In this situation, the company will pay for a charter flight to satisfy one of the demands, which is usually expensive and increases the operating cost. However, if the second owner's flight is requested to depart at 15:00, it may be possible for the fractional airplane to fly from A to B to take the first owner, reposition (dead-head) from B to C, and then fly from C to A to take the second owner. In this case, there's no charter cost, but there is the additional cost of a repositioning flight.

In general, fractional airlines can always reduce the total operating cost by efficiently scheduling available crews and airplanes. The worst case (in term of number of owner-requested flights) most likely will happen around holidays, when most people travel. Therefore, most fractional airlines introduce "Peak Days", in which a flight may depart earlier or later by a greater amount of time than usual and may have different pricing. Most fractional airlines also have pre-paid "Jet Card" programs. For example, customers in the program pay a one-time fee for a 25-hour jet card and can fly up to 25 hours without any additional charges. This type of programs gives customers more flexibilities and less obligations. However, it will make the scheduling problem more difficult, because the number of requested flights will increase, but the number of available airplanes is the same as before. When there is no confusion, we will use the term "customers" instead of "owners" to refer to both customers in the Jet Card program and owners in the fractional ownership program.

Note that a Jet Card program is governed under the Part 135 of Federal Aviation Regulations (FAR) (same as charter airlines), while a pure fractional ownership program is governed under the FAR part 91K. Since airplanes, crews and customers of

Jet Card and fractional ownership programs are not differentiated in the scheduling, fractional airlines usually need to follow both FAR 135 and 91K.

In the following sub-sections, we will introduce optimization problems in the planning for commercial airlines and the Dial-a-flight problem, and their differences from the planning for fractional airlines.

1.1.1 Commercial Airlines

Although planning for fractional airlines and for commercial airlines have some similarities, their difference is fundamental, resulting in two different sets of optimization problems. Planning process for fractional airlines is highly dynamic, with changing availability of resources and demand. And the regulations for the crews and airplanes of fractional airlines are also different. On the other hand, a lot of operations research techniques have been successfully applied to the strategic and tactical planning for commercial airlines (see [18], [1] for surveys and references), and some of these techniques are still applicable in the case of fractional airlines.

Due to the complexity and size of the problems, planning for commercial airlines typically consists of several sequential stages. In the following, we will illustrate briefly the similarities and differences between each stage and its counterpart in the case of fractional airlines.

(a) Schedule design. The task at this stage is to determine the set of all flights. Operations of most major commercial airlines are based on a hub-and-spoke network, in which direct flights are between a hub and a spoke, and passengers may need to connect at a hub. During the flight schedule design process, many factors need to be considered, for example, analysis of the current market, passenger demand forecast, important historical factors and decisions of the upper management etc. For domestic operations, flight schedules are repeated daily with some exceptions. Flight schedules for international operations are

typically weekly. Once the schedule is made, origin, destination and departure time for each flight are fixed. In some cases, departure time may be flexible, within a time window of width up to several minutes.

Fractional airlines operate on a point-to-point network that contains much more airports, generally any available airport in the US, or internationally for larger fractional airlines, subject to airport curfew, airplane type compatibility etc. Moreover, customers of fractional airlines may request a flight as late as a few hours before the departure. Therefore, fractional airlines do not operate on a regular schedule, but provide on-demand service instead. Note that flights are always direct flights (except for necessary fuel stops) in the case of fractional airlines.

In our approach, there are three types of flights: customer-requested flights, repositioning flights and trivial flights that correspond to maintenance or appointments. Origin, destination and departure time of a customer-requested flight are determined by the customer who made the request (there may be a time window of up to several hours for the departure time, depending on the contract with the customer). Repositioning flights are determined by the optimization model, in order to reposition an airplane to the origin airport of the next flight that this airplane is assigned to. Note that there are no customers on a repositioning flight. As inputs to the optimization model, each maintenance request (or appointment) is converted to a trivial flight that has the same origin and destination. For example, if a maintenance request for airplane p needs to be performed at airport A, starting at time t and ending at time $t + \delta$, then a trivial flight is generated with origin A, destination A, departure time t, duration δ and flight time 0. Moreover, this trivial flight is tied to airplane p, meaning that airplane p has to be assigned to this flight in any feasible schedule. Similarly, trivial flights are generated from appointments tied to specific

airplanes.

(b) Fleet assignment. After flight schedules are determined, a fleet type is assigned to each flight leg at this stage. Most major airlines have multiple fleets of airplanes. So given a flight leg and a forecast of number of passengers on this leg, if a small fleet type is assigned, then some passengers may not be accommodated (spilled passengers). If a large fleet type is used, then operating cost of this leg may increase. So at this stage, a fleet assignment model (FAM) is solved to minimize the total cost, subject to available fleets.

Most fractional airlines also have multiple fleet types. However, in this case, a customer is always associated with a valid fleet type, which is either the type of the airplane that this customer owns or the fleet type for which this customer buys a Jet Card. Therefore, a customer-requested flight also comes with a fleet type, which is the fleet type associated with the customer who made the request and is denoted as the requested fleet type of this flight. It is considered acceptable if a customer-requested flight is assigned with a fleet type larger than the requested fleet type of this flight. This is denoted as a fleet upgrade, and there may be a limit on the largest fleet type that a given fleet can be upgraded to. Downgrade is not allowed in our approach in order to ensure customer satisfaction. So in the planning for fractional airlines, fleet assignment is also a problem to be solved.

(c) Aircraft routing. Each flight leg is assigned with a specific airplane at this stage, subject to fleet type requirement on the leg and maintenance requirement on the airplane. Fleet type requirement on each leg is from the FAM solution. Maintenance requirements are company specific or from regulations. In general, once the accumulated flying time, number of landings and other criteria on an airplane have reached to certain amounts, various checks on this airplane must

be performed at a qualified maintenance location to satisfy the maintenance requirement. In the aircraft routing problem, a route is a string of flight legs that starts and ends at a maintenance location, and the objective is to find a set of routes to cover each flight once and satisfy all the maintenance requirements. Note that, prior to this stage, all flight legs have be determined, and fleet type of each flight leg has been determined too. Moreover, for a daily flight schedule, there are no repositioning flights in most cases, due to the airplane flow conservation constraints at each airport in the FAM. Therefore, all feasible solutions to the aircraft routing problem have the same cost, and the routing problem becomes a feasibility problem.

For fractional airlines, maintenance requirements have been converted to trivial flights as inputs to the model. Therefore, we need to cover each flight once, subject to individual requirements of each flight. Due to the point-to-point network and the fact that customers may only request one-way trips, airplane flow conservation at each airport can not be satisfied without repositioning flights. Therefore, different routing solutions have different costs, and reducing total repositioning cost is crucial in improving the profitability of fractional airlines.

(d) Crew scheduling. For commercial airlines, crew scheduling model consists of two sequential modules: crew pairing and crew assignment. In the crew pairing problem, a pairing (or tour) is a string of flight legs that starts and ends at a crew base, and a feasible pairing has to satisfy complex crew rules enforced by the FAA regulations. In a pairing, some legs may be repositioning legs for the crew associated with this pairing. In other words, crew is on these flight legs as passengers. The objective of the crew pairing problem is to find a minimum cost set of pairings that covers each flight leg once, subject to crew rules. Note that no specific crew members are assigned to the pairings, and only the crew base

for each pairing is determined in the crew pairing problem. In the following crew assignment module, specific crews are assigned to each pairing, and a typically monthly schedule for each pilot is generated by either the bid line approach or crew rostering. These individual schedules satisfy the training, vacation and preference requirements of each pilot.

Crew rules for fractional airlines are less strict than crew rules for commercial airlines in general. However, customer-requested flights may be between any two airports, and airplanes may be available anywhere. So crew members usually need to travel by commercial flights to where the airplane is, and travel back home at the end of a tour. However, fractional airline usually use small airports at which commercial airlines have no operations. In order to travel with commercial airlines, crew members need to take ground transportation to and from bigger airports, which means that travel is more time consuming than crew repositioning in the case of commercial airlines. Note that travel may be considered as part of a crew member's duty, so we need not only a feasible commercial flight, but also an optimal one. However, all flights at a given airport may not be explicitly available to the model, so the model may need to query an outside database or make assumptions in order to get desired available commercial flights. Another important difference is that crews in fractional airlines usually work on a 7-days-on-7-days-off basis.

1.1.2 Dial-a-flight Problem

Planning for fractional airlines is also a variant of the vehicle routing problem. Among other variants, dial-a-flight problem (DAFP) is the one most similar to the planning for fractional airlines. Dial-a-flight problem is defined by Espinoza et al. in [10] and [11] and arises from the new air-taxi airlines that operate regionally with a fleet of small jet airplanes. We will introduce and explain some characteristics of the DAFP

and the differences between the DAFP and the planning for fractional airlines.

In the dial-a-flight problem, customers request flights and specify origin, destination and a time window for pick up time. The air-taxi airline may reject a flight request, if the request is considered not compatible or profitable with the company's current available resources and schedules. This decision is made by a heuristic algorithm that runs within seconds. An accepted flight request must be scheduled and may be scheduled to contain at most one intermediate stop. But customers must not change the airplane at an intermediate stop. Note that a customer may share the cabin with other customers on a flight, subject to the capacity and weight constraint of the airplane. Therefore, a key characteristic of DAFP is that customer requests may be combined, so that a single flight leg can serve multiple customers.

Other characteristics of the DAFP are the following: there is a single fleet type; a crew consists of a single pilot; pilots from the same base are homogenous, but airplanes are heterogeneous; DAFP is a daily problem, in which a pilot and an airplane must start from a base and return to the same base after duty ends and before the end of the day; the objective function doesn't include crew related cost; crew rules are modelled by setting two shifts for each crew during a day, letting crews start and end a shift at the base, and translating the limit on the flight time of a crew duty to the limit on the flight time of a shift.

1.2 Our approach and contributions

In our approach to the planning for fractional airlines, the planning horizon is typically one to three days. Longer horizon is possible, but may not help, because some customer requests in the farther future of the planning horizon have not come yet. The set of flights is given as input, including trivial flights that correspond to maintenance and appointments. Each nontrivial flight requires two pilots. Other inputs include crews and airplanes of multiple fleet types. Pilots and airplanes may be available

anywhere at any time (not necessarily available before the planning horizon), and a pilot may have any accumulated duty time and flight time when available. It is assumed that each pilot's monthly work schedule is also given as input, which implies that tour start and end time of each pilot are given as input.

The objective is to find the minimum cost assignment of crews and airplanes to flights, so that each flight is covered once and all the feasibility rules are satisfied. If a nontrivial flight can not be covered, then it is assumed to be covered by a chartered airplane with extra cost. Note that this means that our problem is always feasible. We are not minimizing number of airplanes or crews used in the solution.

One advantage of our approach is that the model can start with airplanes available anywhere at anytime and with crews available anywhere with any accumulated flight time and duty time. This enables our model to run in rolling horizon procedures with any given rolling periods. For example, if an unscheduled maintenance event occurs to an airplane at time t, then there are two cases: the problem is minor, and can be fixed at this airplane's current airport; this airplane may need to fly to an available maintenance facility to be further checked. In either case, the to-be-executed schedule may need to be modified. If this happens, we can generate a dummy flight for the new maintenance task, take a snapshot of all crews and airplanes around time t (meaning that updating the status of all crews and airplanes after the schedule is executed up to time t), and re-run the model with the planning horizon starting from time t to obtain a new schedule. Other unscheduled events can also be handled in this way.

We use a generalized set partitioning formulation to model the problem, and solve it using column generation with a constrained shortest path problem as the pricing subproblem. Our contributions include the followings:

1. Exact handling and generalization of demand time windows. We generalize the demand time window to duties, prove results about the properties of duty time

window and duty crew time window, and show how to obtain these time windows algorithmically and efficiently. Moreover, we generalize time windows to tours, and add them to the MIP formulation to enable more feasible connections between tours;

2. Optimization of all elements of a tour, including duties, arcs, travel to pick up airplanes, pickup arcs and travel to home bases. We show how to optimize these tour elements in details at each step of the preparation and tour generation procedure. Moreover, we prove results for arcs aggregation, and properties of the duty network.

3. Efficient constrained shorted path algorithm in the pricing sub-problem. We show that solving a constrained shorted path problem in the duty network is necessary for our approach, and how the label setting algorithm is used to solve it;

4. Analysis of the tradeoff of running time and solution quality. We study various strategies and assumptions used by the model in practice, and provide restricted versions of our model that still improve over the model in practice, but have comparable running time;

5. Reserve airplanes and crews to deal with recovery problem caused by unscheduled maintenance. We modify our main model and consider the scenario when there is unscheduled maintenance. We let the model reserve airplanes and crews to provide extra coverage for customer requested flights that may be interrupted by unscheduled maintenance.

Our computational results on the actual data from a major fractional airline company show the followings: our main model has an average 3.3 percent total cost reduction over the model used in practice; solution quality of our model is consistent

on longer planning horizon; our recovery model shows improvements in the experiments of simulated unscheduled maintenance;

1.3 Literature Review

Column generation was first introduced by Dantzig and Wolfe [5] and was applied to solve the cutting stock problem by Gilmore and Gomory [13], [14]. It has been used to solve large scale linear and mixed integer programs (MIP) successfully. Many practical problems can be modelled with a generalized set partitioning (or covering) formulation with huge number of columns and solved by column generation. In a typical column generation approach to solve a linear program (LP), we consider a subset of all possible columns at each iteration, and the LP with the subset of columns is denoted by restricted master problem (RMP). And then a pricing subproblem (column generator) is solved to add new columns to the RMP. The process is repeated until an optimal solution is found, or a stopping criteria is met. To solve a MIP, column generations is used in a branch-and-bound frame to obtain lower bound at each node of the search tree. This results in approaches such as branch-and-price and branch-and-price-and-cut, depending on whether cuts are also added.

Column generation can solve large scale problems with very complex or nonlinear constraints, by translating these complicating constraints to the pricing subproblem. Particularly, it has been applied to solve hard optimization problems related to airlines successfully. Our model is also a variant of column generation. Note that while column generation is a very general approach, modelling practical constraints, constructing and solving the pricing subproblem, speed of the algorithm and memory management etc. are problem specific and challenging. (see Lubecke and Desrosiers [20] for a recent survey on this topic).

In our model, there are time windows for flight departure times. The simplest example of routing or scheduling problems with time windows is the elementary shortest

path problem with time windows (ESPPTW). In ESPPTW, each node i in the given directed graph is associated with a time window $[a_i, b_i]$, and the time windows for two terminal nodes, s and t, are trivial, i.e., $a_s = b_s$ and $a_t = b_t$. Each arc ij has a positive duration d_{ij} and arbitrary cost. A feasible s-t path P starts from node s at time a_s and ends at node t before time b_t, and there exists a set of times $T = \{a_i \leq t_i \leq b_i : \forall i \in P\}$ such that $t_i + d_{ij} \leq t_j$ for any arc $ij \in P$. The objective is to find a minimum cost feasible path. ESPPTW is strongly NP-complete, therefore no pseudo-polynomial time algorithm is possible. If the path is allowed to contain cycles, then the resulting problem is still NP-complete. But it can be solved in pseudo-polynomial time by discretizing the time windows to construct an acyclic network or by dynamic programming methods. ESPPTW is a special case of the shortest path problem with resource constraints (SPPRC). Note that the pricing subproblem of column generation approach for general routing or scheduling problems is usually a SPPRC problem. See Desrosiers et al. [7] for general methods to solve SPPRC and related problems.

In the following subsections, we will review some works in the related areas such as commercial airlines, charter airlines etc., which study problems similar to ours or use techniques similar to our approach, and review past works in the planning for fractional airlines.

1.3.1 Commercial Airlines

Extensive amount of research has been devoted to the planning for commercial airlines and the literature are rich (see Barnhart et al. [1], Barnhart et al. [2] and Klabjan [18] for surveys and references). In particular, approaches in which multiple planning stages are integrated and there are time windows for flights are related to our problem. Rexing et al. [23] consider the fleet assignment model (FAM) with time windows. Desaulniers et al. [6] consider time windows for flights and integration of

fleet assignment and aircraft routing. Mercier and Soumis [22] and Klabjan et al. [19] consider small time windows for flights and integration of crew pairing and aircraft routing. For commercial airlines, when the departure time of a flight changes, demand for this flight may change. Therefore, the assumption that demand within the time window doesn't change must be made. In addition, time window of a flight is always centered at the planned departure time.

In Rexing et al. [23], each flight has a time window less than forty minutes. The basic FAM, a multi-commodity flow formulation in which each commodity corresponds to airplanes a fleet type, is based on the flight networks. There is a flight network for each fleet type. And in a given flight network, there is a time line for each airport and there are flight arcs between time lines. Suppose that we have built the flight networks according to the planned departure time of each flight. Starting from these networks, if we add a time window to a flight originally having no time windows, we can replace the corresponding original flight arc with one of the followings: *relaxed arc*, a single flight arc that departs at the end of its departure time window and arrives at the beginning of its arrival time window (therefore the duration of a *relaxed arc* is reduced by the length of its time window); *discretized arcs*, a set of arcs, each of which is a copy of the original flight arc and corresponds to a discretized interval of the time window. Time lines are updated accordingly if *relaxed* or *discretized* arcs are added.

Two approaches are proposed and compared in [23]: the first approach is to use *discretized arcs* to replace each original flight arc that corresponds to a flight with time windows, and then use a solver to solve the FAM based on the new flight networks directly; the second approach is an iterative procedure between a master problem and a subproblem. The master problem is the basic FAM, initialized by using a *relaxed arc* to replace each original flight arc that corresponds to a flight with time windows. At each iteration of this procedure, the master problem is solved. The

solution to the master problem is then processed by the subproblem to identify a set of "*problem*" flight arc pairs. *Problem* arcs must be *relaxed arcs*, and each pair of *problem* arcs has the connection that is infeasible for the corresponding actual flights, but feasible for the master problem, because they are relaxed in the master problem. The master problem is then updated by replacing all *problem* arcs with *discretized arcs*, and the iterative procedure continues. Computational experiments are run on practical instances with up to 2000 flights, 11 fleet types and time windows of 20 or 40 minutes. Results show that the total cost can be reduced by the range from 0.3 percent to 0.7 percent, which translate to daily savings between $65,442 and $126,553.

Desaulniers et al. [6] consider the integration of fleet assignment and aircraft routing, but crews are not considered in [6]. The problem is to route a set of heterogeneous airplanes for given a given set of flights (with time windows) in a one-day planning horizon. Although there are no depots for airplanes, number of airplanes of any fleet type at any airport must be the same at the beginning and at the end of the day. Two equivalent models are proposed: a set partitioning formulation, in which each column corresponds to an airplane route, and a compact multi-commodity flow formulation, in which each commodity corresponds to airplanes of a fleet type. The models are solved by column generation and branch-and-bound that branches on flow variables of the multi-commodity flow formulation. The pricing subproblem is a longest path problem with time windows and is solved by dynamic programming under the assumption that the time windows are narrow and the underlying network is acyclic. Computational results based on two sets of practical data are the following: in the first set that has time windows less than 60 minutes, with the objective to minimize fleet size first and maximize profit second, the fleet size can be reduced by 14% comparing to no time windows; in the second set, the time windows are less than 20 minutes, and the total number of available airplanes is fixed, but the model decides the number of airplanes of each fleet type. For the second set of data, the total profit

is improved by more than 10%, comparing to the solution provided by the airline that has a pre-fixed fleet composition and no time windows.

The crucial point in integrating crew pairing and aircraft routing for commercial airlines is the following: the minimum time for airplanes to connect between two flights (min turn time) is usually smaller than the minimum time for crews to connect (min sit time), but min sit time can be reduced if crews don't switch airplanes in a connection. Moreover, the aircraft routing problem is a feasibility problem after the FAM is solved, therefore crew pairing problem should be emphasized in order to reduce the total cost of these two problems. The interaction between pairing and routing, plus the emphasis on the crew pairing, are utilized in both [19] and [22]. In particular, [22] presents a full integration, and [19] considers the crew pairing problem with additional constraints that ensure feasible aircraft routings afterwards.

In Klabjan et al. [19], the crew pairing problem is modelled with a set partitioning formulation, in which each column corresponds to a crew pairing with min sit time set to be equal to the min turn time. Therefore, a solution (crew pairings) to this formulation may contain forced turns that have crew-connecting time less than min sit time. A forced turn forces the corresponding pair of crew-connecting flights to be be assigned with the same airplane. Therefore, forced turns restrict airplane connections and may require more airplanes on the ground than the FAM solution. The result is that the following aircraft routing problem may be infeasible. So plane-count constraints are added to the formulation to ensure that, for each ground arc, the number of airplanes required by forced turns associated with this ground arc is less than or equal to the number of airplanes of this arc in the FAM solution. All columns are generated up front, and the model is solved by a branch-and-bound method.

Let us illustrate how the columns (pairings) with time windows are generated in [19]. WLOG, let us assume that the time window for each leg is the same and equals to w. A set of potential parings (not all) is generated according to the *relaxed*

crew feasibility rules due to time windows. For instance, the min sit time is reduced by $2w$. Each potential pairing is then re-timed to form an "utmost left re-timing", in which each leg in the pairing departs as early as possible, subject to the time window of each leg and *actual* crew feasibility rules. A potential pairing is discarded if it can not be re-timed. Note that the utmost left re-timing process does not try to minimize the duration of a duty or a pairing, and it returns a single feasible timing for a pairing. All remaining potential pairings in the set are feasible and will be the columns of the set partitioning formulation. Plane-count constraints are added, except for ground arcs with length less than w. For these arcs, an incoming flight and an out-coming flight may be swapped during the re-timing process, resulting in infeasibility due to higher plane count. This can be handled with additional processes, and the authors report that it rarely happens in the computational experiments. Computational results on the data with up to 450 legs and with time windows less than 20 minutes show that the total crew cost can be reduced substantially ,with running time up to 10 hours and comparing to the traditional routing-first-pairing-second approach. In particular, cost is reduced by up to 25% comparing to the same model but without time windows.

In Mercier and Soumis [22], each leg has three possible departure times: the planned departure time and ± 5 minutes. The model is also a set partitioning formulation, in which there are two types of columns: crew pairings and aircraft routes. Additional constraints enforce the following: for each leg, the same departure time is chosen for aircraft, crew and repositioning crews (crews who take a flight as passengers); for each short connection arc, the same aircraft is assigned. The model is solved by a Bender's decomposition method that has the crew pairing as the master problem and the aircraft routing as the subproblem, and it uses column generation at each iteration. The model, run on a set of practical data with up to 500 legs, can have a significant speedup after aggregation and dynamic generation of the additional constraints mentioned above. Aside from the cost savings of integrating crew pairing

and aircraft routing, it's shown that the total crew cost can be reduced by 1.6% and number of aircrafts can be reduced by 2, comparing to the same model but no flight leg re-timing.

1.3.2 Pickup and Delivery Problem (PDP), Dial-a-flight Problem (DAFP) and Charter Airlines

A recent survey about the well studied General Pickup and Delivery Problem (GPDP) is Savelsbergh and Sol [26]. In the GPDP, a customer specifies an origin and a destination and requests goods or people to be transported by a vehicle from the origin to the destination. Vehicles have capacities and vehicle routes may be subjected to time constraints. Variants of the GPDP are reviewed in [26]. In particular, Dumas, Desrosiers and Soumis [8] consider a variant in which there are time windows for the pickup time of customers. A column generation approach with constrained shortest path subproblem is used in a branch-and-bound frame to solve the problem. For each vehicle, the pricing subproblem is to find the minimum reduced cost route that respects time windows, priority and capacity constraints. Various ideas are discussed to reduce the network size in the subproblem, to eliminate labels in the dynamic programming algorithm that solves the subproblem, and to speed up the algorithm. Instances of up to 22 vehicles and 55 requests with different characteristics are solved successfully.

Dial-a-flight Problem is defined and studied in Espinoza et al. [10] and Espinoza et al. [11]. In [10], a multi-commodity flow model is used, in which each commodity corresponds to a single airplane. The planning horizon is denoted by $[0, 1440]$, i.e., minutes of a day. A time-activity network is constructed for each airplane. More specifically, for each airplane and each airport, a subnetwork is constructed to model all possible activities of this airplane at this airport at each discretized time point in $[0, 1440]$. Note that there is a discretization of $[0, 1440]$ for each airplane at each

airport. Activities at any given time include: idling on the ground; departing to another airport to satisfy one or more requests by using only this flight (direct loading); departing to another airport to satisfy one or more requests by using this flight as one of the two flights (indirect loading). Flight arcs connect subnetworks of different airports. Nodes and arcs are aggregated to reduce the size and improve the LP bound. And the time discretization for each airplane at each airport can be $[0, 1440]$ discretized to any level, and it has two sides: finer discretization to get better solutions and coarser discretization to reduce the network size. Note that at any discretizing level, a set of special time points are always added to ensure possible good solutions are not cut off by the discretization. In the network constructed for an airplane, a unit flow thus corresponds to a candidate route for this airplane and has all the necessary information: flights, departure times and passengers of each flight etc.. Side constraints are also added to ensure capacity, weight and other airplane specific constraints.

The model is solved directly by a solver. Computational results are based on three sets of practical data, which have 4 to 8 airplanes and 43 to 81 requests and are provided by the airline. Each data set consists of 23 instances with the same fleet. The results show that the total cost (in term of total flying time) can be improved by 6 percent to 15 percent, comparing to the solution returned by the heuristics that the airline use to determine whether a request will be rejected. The results also show a seemingly counterintuitive observation that finer discretization returns much worse solutions. This is because that special time points are always added to the discretization despite of the discretizing level and that the larger size of the model with finer discretization prevents a good solution from being found within the given running time limit that is set to be 4 hours.

Much larger instances (with about 300 airplanes and 2,800 request) are considered in [11]. Multi-commodity flow model in [10] is not able to solve problems of this

size, but it is used as building blocks in a parallel local search approach with the objective to get the best improvement within 4-hour running time, starting from a given feasible schedule. At each step of the local search, a small subset of airplanes are chosen. The subproblem, induced by the chosen airplanes and requests served by these airplanes in the current best schedule, is solved by the multi-commodity flow model. Several metrics and diversification strategies are combined with asynchronous parallel computing on 16 processors to form different schemes. Computational results for all the schemes, based on a set of data containing 10 instances, show the improvements range from 5.7 percent to 9.7 precent, comparing to the solutions returned by the airline's heuristics.

There are two types of operations for charter airlines: on-demand, in which customers request flights and specify the origin, destination and departure time of each desired flight, and customers do not share the cabin; scheduled, in which schedules repeat weekly, and customers book round trips and specify the origin, destination, departure date and return date of each round trip. Airplanes in the on-demand case are usually smaller than airplanes in the scheduled case. In the on-demand case, the optimization problem can be reduced to a full-truckload PDP (Savelsbergh and Sol [26]). Ronen [24] considers a different approach for the planning for a special case of on-demand charter airlines. The scheduled case is motivated by vacation or leisure travellers and is studied by Kim and Barnhart [17] and Erdmann et al. [9]. Note that, in the scheduled case, a one-way trip can be satisfied by a *direct* flight, or a *via*-flight that has an intermediate stop. In contrast to the DAFP, both [17] and [9] assume that whether a *via*-flight is allowed for an origin and destination pair is given by the airline as inputs to the optimization model, which simplifies the underlying optimization problem.

The planning horizon in Ronen [24] is between 24 and 48 hours. An explicit set partitioning formulation, in which each column is a feasible airplane route (sequence

of flight legs, including repositioning legs), is used to model the problem. There are penalties for idle airplanes, and crew related rules are not discussed. However, a crew swap is counted in the case when a route can not be completed within one crew duty, and there is an upper bound on the total number of crew swaps of all routes. For each airplane, a subset of all feasible routes is generated up front, using a breadth-first search on all requested flights. In order to control the total number of generated routes, only k closest neighbors of a requested flight are explored in the search, where k is a pre-specified parameter. The model is then solved by an IP solver directly. Computational results on the data set with up to 50 aircrafts and up to 100 requested flights show that the model can be solved to near optimality within minutes.

In the case of scheduled charter airlines, flight schedules are assumed to repeat weekly, and demand (number of passengers) for each origin-destination ($o-d$) pair is assumed to be symmetric on any given day, i.e., the number of passengers from o to d is equal to the number of passengers from d to o on any given day. This is motivated by the observation that vacation travellers who departs on a given day usually return exactly one week or several weeks later. There may be multiple flights with the same origin and destination on a given day, in order to provide enough capacity (number of seats). Departure time of each flight leg and the assignment of passengers to flight legs are decided by the airline. In other words, customers are concerned about the date, not the time. Daily rotation of an airplane, route of an airplane that starts and ends at the same airport, must have the following property: if a rotation provides capacity equal to c from o to d, then it must also provide the same capacity from d to o, and vice versa. Moreover, there is no repositioning (dead-heading) between flights. Therefore, flights legs of a rotation can be partitioned into a set of round trips. If there is an upper bound on the total flying time of a rotation, it is checked approximately by limiting number of flights in the rotation.

In Kim and Barnhart [17], at most three flights can be used to satisfy the demand between an $o-d$ pair. A daily network, in which each node corresponds to a round trip, is constructed for each day of the week. Additional restrictions on rotations ensure that there is a one-to-one correspondence between a path in the network (a sequence of round trips) and a feasible rotation. And then seven daily networks are linked together, since airplanes may be repositioned to other locations at the end of each day. The profit maximizing problem is formulated as a network flow problem and solved directly by an IP solver. Computational results on a set of practical data show that solutions with the optimality gap of 4.5 percent can be found within one hour of running time. A heuristics algorithm that decomposes the problem according to fleet types in order to reduce the running time is also discussed.

In Erdmann et al. [9], there are two types of airports: home and abroad airports, and customers book trips between a home airport and an abroad airport. An airplane rotation must start from a home airport and contain at most two *via*-flights. If a rotation contains a single *via*-flight, then this *via*-flight must be via an abroad airport. If a rotation contains two *via*-flights, then they must be a pair of symmetric *via*-fights via the same home airport. A passenger itinerary is a one-way trip that can contain at most two flight legs. In the MIP formulation, columns correspond to airplane rotations and passenger itineraries. Although the pricing subproblem is discussed, all columns are enumerated up front to get an explicit formulation. This is possible because of moderate size of the problem and the assumptions that restrict the airplane rotations. Branching rules and generalized flow cover inequalities are discussed to adopt a branch-and-bound approach to solve the MIP. Based on a data set provided by an European charter airline with up to 18 home airports, 40 abroad airports, 26 airline-owned aircrafts and 90 $o-d$ pairs, computational results show that the cost can be reduced by up to 42 percent, comparing to the solutions provided by the airline that are calculated by hand, and the optimality gap is a few percent.

1.3.3 Fractional Airlines

Literature about planning for fractional airlines begin with Keskinocak and Tayur [16]. Some properties of the problem defined in [16] include: crews are not considered; there is a single aircraft type; departure time of each flight is fixed; the planning horizon is less than 3 days. The optimization problem is an aircraft routing problem with two additional complicating constraints: pre-scheduled flights, each of which must be scheduled to a specific airplane, and maintenance restrictions, which enforce limits on total flying hours and number of landings of an airplane within the planning horizon. The objective is to minimize the total operating and charter cost. The restricted case with either of the additional constraints is shown to be NP-Complete. An IP formulation, equivalent to a multi-commodity network flow model on the flight network in which each commodity corresponds to an airplane, is constructed and solved by an IP solver directly. Several heuristics algorithms are also discussed and tested. Computational results on a set of randomly generated data show that the heuristics algorithms can find near optimal solutions within seconds, much faster than solving the IP directly.

In Martin et al. [21], the IP formulation in [16] is extended to include crew constraints and multiple aircraft types. The planning horizon is 2 or 3 days. Customers can be upgraded to a larger airplane, but can not be downgraded. For each airplane, a pair of pilots are assumed to be available at the same location and time as the airplane. No crew swaps are allowed in the schedule. In the IP formulation, besides all constraints in [16], additional constraints are added to ensure that the time between any two possible consecutive rests in an airplane route is less than 14 hours, which implies the maximum duty time of 14 hours for any duty. The maximum flight time between rests and the 10-in-24-hour rule are not considered. The complete approach implemented for a major fractional airline company shows 18.7 percent reduction of operating cost in one year.

Along the lines of Keskinocak and Tayur [16] and Martin et al. [21], two new approaches are considered by Yang et al. [27]. In their first approach, an integer network flow formulation is constructed to model the aircraft routing problem that has multiple aircraft types but no maintenance restrictions. After the model is solved directly by the CPlex, the returned solution (aircraft routes) is checked by a heuristic to ensure the maximum duty time constraint. The second approach is a set partitioning formulation, in which each column is an aircraft route that also satisfies the maximum duty time constraint. It is solved by a branch-and-price method. In the computational results based on both randomly generated and real data, the authors report that the first approach is more efficient with short planning horizon (24 hours), while the second approach is more efficient with longer panning horizon and more complicating constraints. In particular, solutions to instances with 48 hours of planning horizon can be found by the second approach in less than 15 minutes and have optimality gap of a few percent.

A complete approach for a major fractional airline company is discussed at high level in Hicks et al. [15]. The size of their problem is similar to ours. Their complete model includes three sequential modules. The *first module* returns a monthly schedule for crews, which includes on-duty days, vacations, training etc. for each crew member. In our approach, we assume that a similar module is present and has been executed, so that monthly schedules for all crews are available as inputs. The *second module* assigns customer requested flights to airplanes and also generates crew pairings. To solve the problem, they use Dantzig-Wolfe decomposition and column generation method, in which each column corresponds a specific airplane, a pair of homogenous crew members and a set of flights. Each column also specifies the base airport from which the crew come, crew resting periods and crew swaps in the set of flights if applicable. The *third module* solves the crew assignment problem and assigns specific crews to the crew pairings returned by the *second module*, with the objective of

minimizing crew travel costs, overtime costs etc. The *third module* is solved by an explicit set partitioning model.

Comparing [15] with our approach, there are differences and similarities in both modelling and algorithmic aspects. More specifically, differences include: we integrate the *second* and *third modules* in [15], while dealing with some practical constraints differently; time windows are discretized into slots of several minutes in [15] and an copy of flight arc is made for each time slot, while we have an exact method for the time windows; there are penalties for the waiting time in [15]; pricing subproblem is for each airplane and is based on the flight network in [15], while our pricing subproblem is based on the duty network (therefore acyclic) and columns are generated for each customer-requested flight; there may be an upper bound on the fly time of an airplane in the planning horizon in [15], while airplanes are aggregated to reduce the symmetry in the formulation in our case, and there is no bound on the flying time (bound can be enforced, if the aggregation of airplanes is removed).

Similarities between [15] and our approach include: size of the instances; length of the planning horizon (less than 3 days); maximum length of the time windows (6 hours).

Aircraft scheduled maintenance is crucial for an efficient schedule. Let's note two different approaches to handle maintenance requirements. One approach is to assume that, for each airplane, whether to perform the maintenance task to satisfy a maintenance requirement on this airplane, and the time and location of performing the task are decided by the model, subject to capacities of a given set of available maintenance facilities. This approach is used in Hicks et al. [15], although no further details about how these decisions are incorporated into the model are discussed.

We use another approach and assume that whether to perform a maintenance task on an airplane and the location and time of performing it are decided before the current planning horizon and are inputs to the model. This assumption is based

on the following considerations: there are several types of maintenance requirements, and their durations and requirements for mechanics and facilities are different; maintenance facilities, especially outside contracted facilities, may need to be contacted before the current planning horizon to make appointments; actual duration and cost of performing a maintenance task may depend on the facility and the time to start the task; the planning horizon is relatively short (less than 3 days), so that the maintenance requirements that are based on accumulated quantities on an airplane (flying time, number of landings etc.) can be approximated reasonably.

Our model can be modified to use the first approach as follows. Aggregation of airplanes needs to be removed, so that each individual airplane, therefore its flying time, number of landings etc., is tracked. Let us assume that the duration of a maintenance task is always a multiple of an integer t, and assume that there is an open time interval for each maintenance facility that is a multiple of t . In other words, duration of a maintenance task and available time of a facility are discretized into time slots of length t.

For each possible maintenance task on each airplane and before the optimization, we find the set of facilities capable of performing this task, and then, for this task starting from each available time slot at each feasible facility, we generate a maintenance request and calculate its duration and cost. Therefore, a maintenance request contains location and starting time slot of performing the corresponding maintenance task. Additional constraints are added to the formulation to ensure that only one of all the requests generated for a maintenance task is in the schedule, and that all tasks performed at a facility are within the capacity of the facility. In the column generation process, a column of an airplane must also satisfy maintenance requirements on this airplane, with proper maintenance requests included in the column.

A similar column generation based approach is proposed in Yao et al. [28]. A set partitioning type of formulation is used, and the pricing sub-problem is based on

the duty network and is for each crew. Computational results in [28] show that near optimal solutions can be found for instances from the actual data.

Differences between the model in [28] and our model include the followings:

1. In [28], some crews are tied to an airplane at the beginning of the current planning horizon, and these crews can only use the tied airplanes during current planning horizon. More specifically, a crew is not tied to an airplane, i.e., they can pick up any available airplane, only in the following cases: they will start their 7-day working period in the current planning horizon; their previous flight was a long maintenance request, therefore they were released or un-tied from their previous assigned airplane.

   In our model, we do not tie airplanes to crews, and any crew can pick up any available airplane.

2. In [28], an airplanes is dropped off after the first tour (a tour corresponds to a column in the MIP formulation), and is picked up by the second tour only if the first tour's crew finished their 7-day working period and must travel to their base airports. In our model, we do not have this assumption, i.e., whether and when an airplane is dropped off are decided by the model.

3. An airplane can only be dropped off after a customer-requested flight in [28]. In our model, airplane can also be dropped off after a repositioning leg.

4. All feasible duties not containing maintenance/appointments are generated in [28]. Then a duty network consisting of these duties is constructed for each crew. Given a maintenance/appointment, if its airplane is already tied to a crew, then this maintenance/appointment is inserted into the tied crew's duty network, so that any path in this network that corresponds to a feasible column must contain this maintenance/appointment. If its airplane is not tied to a crew

yet, then the model in [28] finds a crew nearest to this airplane and ties them before optimization.

In our model, we enumerate duties that contain maintenance/appointments, but we only enumerate duties that start and end with a demand, and we include the first or/and last repositioning leg in the arcs. There is a duty network for each fleet type, and we do not manually tie airplanes that will have maintenance to crew. Moreover, maintenance/appointments are not mandatory in our model, i.e., our model has the choice to skip an appointment by paying the corresponding penalty.

5. In [28], airplane connections between two tours are restricted, i.e., if an airplane has a few maintenance/appointments during the current planning horizon, then it is not feasible that two tours use this airplane, and each tour contains maintenance/appointments of this airplane. This case is feasible in our model.

6. A duty is fixed with a timing, and this timing can not be changed during the optimization in [28]. In our model, duty has a time window instead of only one fixed timing.

7. In [28], pricing subproblem is a shortest path problem. In our model, pricing subproblem is a constrained shortest path problem solved a label setting algorithm.

8. Optimization of elements of a tour is not presented in [28]. In our model, we present our objective and how to optimize with respect to that objective in details and at each step of constructing our model.

In our computational tests, we compare our model to a model used by a major fractional airline company in practice, which is based on and improves over the model in [28]. More specifically, the model used in practice and our model deal with the

above difference 1 though 5 in the same way (except for generating duties and arcs), while the model used in practice and the model in [28] deal with the above difference 6 through 8 in the same way.

In other words, our model generalizes the model used in practice, which generalizes the model in [28]. Moreover, the model used in practice and our model also differ in many other aspects, and this will be discussed in details in Section 4.3.2.

CHAPTER II

NOTATION, DEFINITIONS AND COMPLEXITY

In this chapter, we will introduce necessary notation and define the optimization problem that we aim to solve at the hight level, And we will prove some complexity results for our optimization problem.

2.1 Notation and Definitions

Let $\mathbb{H}$ be the planning horizon, usually less than or equal to three days. $\mathbb{H}$ may start and end at any time during a day. The available time and location of each airplane operated by the fractional airline are given, so that $\mathbb{A}$, the set of airplanes that are available before the end of $\mathbb{H}$, is well defined.

Let $\mathbb{F}$ be the set of fleet types of all airplanes operated by the fractional airline. Note that airplanes of different fleet types have different operating costs and other characteristics. It is possible that no airplanes of a particular fleet type is in $\mathbb{A}$, i.e., all airplanes of this fleet type are available after the end of $\mathbb{H}$, but this fleet type is still defined in $\mathbb{F}$, so that an *order* of $\mathbb{F}$ and *upgrade* are well defined.

The *order* of $\mathbb{F}$ is defined such that if $f_1, f_2 \in \mathbb{F}$ and $f_1 < f_2$, then airplanes of fleet type f_2 are larger and "better" than airplanes of fleet type f_1, and if we assign an airplane of fleet type f_2 to a customer who requested a flight with an airplane of fleet type f_1, it is feasible and is considered as an *upgrade* for the customer.

For $f_1, f_2 \in \mathbb{F}$, let $f_2 - f_1$ be the difference of positions of f_2 and f_1 in the order. For example, if there is only a single fleet type greater than f_1 and less than f_2 in the order, then $f_2 - f_1 = 2$. Let $nu \in Z^+$ and $nu \leq |\mathbb{F}|$ be the maximum number of *upgrades*. An *upgrade* from f_1 to f_2, $f_1 < f_2 \in \mathbb{F}$, is defined to be feasible, if $f_2 - f_1 \leq nu$. *Downgrade* is not allowed.

Let $a \in \mathbb{A}$. Attributes if airplane a include the following:

- $aa(a)$: available airport of a;
- $at(a)$: available time of a;
- $fp(a)$: fleet type of a.

Information about a pilot includes available time and location, base location, tour start time and tour end time. Let $\mathbb{C}$ be the set of pilots available before the end of $\mathbb{H}$. Each pilot is assumed to be associated one fleet type. In case that a pilot can operate airplanes of multiple fleet types, we can make a copy of this pilot for each feasible fleet type and require that only one of the copies can be used in a feasible schedule.

For duties within the current planning horizon, we do not assume that a pilot must start from a base or start after rest. A pilot may already have accumulated duty time or block time when available, and a pilot may be available anywhere at anytime. But a pilot must start her/his tour (usually a week) from the base at her/his tour start time and must end the tour at the base before her/his tour end time.

However, the tour start or end time is not a hard constraint. Each pilot indicates how many days the tour can be started earlier, how many days the tour can be ended later, and whether there is extra cost for the overtime. This extra cost is defined as the *overtime cost*.

Let $c \in \mathbb{C}$. Attributes if pilot c include the following:

- $aa(c)$: available airport of c;
- $at(c)$: available time of c;
- $fp(c)$: fleet type of c;
- $ba(c)$: base airport of c;
- $ts(c)$: tour start time of c, not including possible overtime;

- $te(c)$: tour end time of c, not including possible overtime.

Let $\mathbb{CP}$ be the set of crew pairs (a pair of pilots including a captain and a first officer). Crew pairs in $\mathbb{CP}$ may not be disjoint and are given as input. Moreover, two pilots in a crew pair may be available at different locations and at different times, so they will need to travel (with commercial airlines) to form the crew pair and pick up an airplane.

Let $\mathbb{D}$ be the set of all demands (flights). $\mathbb{D} = \mathbb{D}_c \cup \mathbb{D}_m \cup \mathbb{D}_a$, in which $\mathbb{D}_c$ is the set of all customer-requested flights that can depart within the planning horizon $\mathbb{H}$, $\mathbb{D}_a$ is the set of trivial flights generated for all appointments that can start within $\mathbb{H}$, and $\mathbb{D}_m$ is the set of trivial flights generated for all maintenance requests that can start within $\mathbb{H}$.

Let $d \in \mathbb{D}$. Attributes of demand d include the following:

- Requested fleet type: $fp(d)$. If $d \in \mathbb{D}_c$, then it is the fleet type associated with the customer who made the request, otherwise there's an airplane associated with d and $fp(d)$ is the fleet type of the associated airplane.

- Origin and destination airports: $oa(d)$ and $da(d)$. They are not the same if $d \in \mathbb{D}_c$, otherwise they must be the same for trivial flights.

- Feasible fleet types. Fleet type $f \in \mathbb{F}$ is not feasible for demand d if any one of the followings holds:

  - f is not feasible for $oa(d)$ or $da(d)$. For instance, runway of the airport is too short for airplanes of fleet type f.

  - f is a *downgrade* of the requested fleet type $f(d)$.

  - f is an *upgrade* of the requested fleet type $f(d)$, but $f - f(d) > nu$.

  - d is a trivial flight and f is not the requested fleet type of d, i.e., $d \notin \mathbb{D}_c$ and $f \neq f(d)$.

Moreover, we assume that the requested fleet type, $fp(d)$, is always feasible for demand d, because the feasibility of $fd(d)$ has been checked when this flight is requested.

- Number of passengers if $d \in \mathbb{D}_c$. This affects the flight time and operating cost of this demand.

- Let f be a feasible fleet type for d. The followings are attributes of d that are associated with the assignment of an airplane of fleet type f to d.

  - $rd_f(d)$: requested departure time of d.
  - $ed_f(d)$: early departure time of d.
  - $ld_f(d)$: late departure time of d.
  - $[ed_f(d), ld_f(d)]$: time window for the departure time of d. Time window is *trivial* if $ed_f(d) = ld_f(d) = rd_f(d)$.
  - $tt_f(d)$: turnaround time for an airplane of type f after demand d.
  - $ft_f(d)$: flight time of d. $ft_f(d) = 0$ if $d \notin \mathbb{D}_c$. Note that if $ed_f(d) \neq ld_f(d)$, we need to assume that the flight time of d does not depend on the departure time of d, i.e., flight time is equal to $ft_f(d)$ as long as the departure time is in $[ed_f(d), ld_f(d)]$.
  - $du_f(d)$: duration of d. $du_f(d) = ft_f(d)$ if $d \in \mathbb{D}_c$. Otherwise, $du_f(d)$ is the duration of the corresponding maintenance request or appointment.
  - $et_f(d)$: elapsed time of d. $et_f(d) = du_f(d) + tt_f(d)$.
  - $ea_f(d)$: the early arrival time of d. $ea_f(d) = ed_f(d) + et_f(d)$.
  - $la_f(d)$: the late arrival time. $la_f(d) = ld_f(d) + et_f(d)$.
  - $c_f(d)$: cost of flying demand d. Note that flight time, $ft_f(d)$, does not depend on the actual departure time. Moreover, there is no cost for waiting

time, and cost of a demand in the objective function is based on fleet type, flight time and number of passengers of this demand. Therefore, cost of demand d doesn't depend on actual departure time of d and is well defined.

The above notation are also defined and have the same meanings in the case when we consider a repositioning flight leg d, except for $rd_f(d)$. $rd_f(d)$ is not defined for repositioning flight, since there is no requested departure time for repositioning flights.

- Bundled before and after demand: $bb(d)$ and $ba(d)$. $bb(d) = \hat{d}$ implies that $\hat{d}$ and d must be assigned with the same airplane and pilots, and $\hat{d}$ must be immediately proceeding d in the schedule. $ba(d)$ is defined similarly.

- Set of *type-feasible* airplanes for demand d. If $d \notin \mathbb{D}_c$, then this set consists of only one airplane. Otherwise, it implies that demand d must be assigned with one of these *type-feasible* airplanes in any feasible schedule. The default set of *type-feasible* airplanes for demand $d \in \mathbb{D}_c$ is the set of all airplanes with fleet type feasible for d.

Let $d_i, d_j \in \mathbb{D}$ and let $f \in \mathbb{F}$ be a feasible fleet type for both d_i and d_j. Let us assume that $da(d_i) \neq oa(d_j)$, the same airplane of fleet type f is assigned to d_i and d_j, and d_i precedes d_j in the schedule. It follows that the airplane needs to be repositioned from airport $da(d_i)$ to airport $oa(d_j)$.

This non-trivial repositioning flight leg is denoted as $d_i d_j$. A repositioning leg makes sense only when there is enough time for the flight to take place after the arrival of d_i and before the departure of d_j. Attributes of a feasible repositioning leg $d_i d_j$ include the following:

- $oa(d_i d_j)$: origin airport of $d_i d_j$. $oa(d_i d_j) = da(d_i)$.

- $da(d_i d_j)$: destination airport of $d_i d_j$. $da(d_i d_j) = oa(d_j)$.

- $tt_f(d_id_j)$: turnaround time after flight d_id_j. This is usually set to be the same for all repositioning legs of the same fleet type.

- $ft_f(d_id_j)$: flight time of d_id_j. Similar to demands, we need to assume that the flight time of d_id_j does not depend on the departure time of d_id_j.

- $du_f(d_id_j)$ and $et_f(d_id_j)$: duration and elapsed time of d_id_j. $du_f(d_id_j) = et_f(d_id_j) = ft_f(d_id_j) + tt_f(d_id_j)$.

- $ed_f(d_id_j)$: early departure time of d_id_j. $ed_f(d_id_j) = ea_f(d_i)$.

- $ld_f(d_id_j)$: late departure time of d_id_j. $ld_f(d_id_j) = ld_f(d_j) - du_f(d_id_j)$.

- $[ed_f(d_id_j), ld_f(d_id_j)]$: time window for the departure time of d_id_j. Note that if $ea_f(d_i) + du_f(d_id_j) > ld_f(d_j)$, then d_id_j is *infeasible*, since it is not possible to arrive before the latest departure time of d_j. Otherwise, $ea_f(d_i) \leq ld_f(d_id_j) = ld_f(d_j) - du_f(d_id_j)$, i.e., the time window is well defined.

- $ea_f(d_id_j)$: the early arrival time of d_id_j. $ea_f(d_id_j) = ed_f(d_id_j) + du_f(d_id_j)$.

- $la_f(d_id_j)$: the late arrival time. $la_f(d_id_j) = ld_f(d_id_j) + du_f(d_id_j) = ld_f(d_j)$.

- $c_f(d_id_j)$: cost of the flight leg d_id_j. Similar to demands, cost of a repositioning leg d_id_j does not depend on actual departure time of d_id_j and is well defined.

A duty $S = (s_1, s_2, ..., s_m)$ is a sequence of flights in the increasing order of departure times. Moreover, for $1 \leq i \leq m$, s_i is demand in $\mathbb{D}$ or a repositioning flight leg, and for $1 \leq i \leq m-1$, $da(s_i) = oa(s_{i+1})$. There are no consecutive repositioning legs in S, since it does not make sense in practice.

Note two things about this definition of a duty: any flight in S may be a repositioning flight (the first or the last or anywhere in between); any flight in S may be a trivial flight. It follows that a duty may start with a repositioning leg and/or end with a repositioning leg. In addition, if s_1 is a trivial flight, then S and $S \backslash s_1$ have

the same set of tasks for crews, but both of them are kept for the easy presenting and handling of other parts of the model.

Duty S is defined to be a *crew-duty* if it contains at least one non-trivial flight. Otherwise, S is a *no-crew-duty*.

In order to define attributes of a duty S, let us consider a special case in which the time window of each demand in S is trivial. We will first define attributes of a duty in this special case, and then in later sections, we will show that theses attributes are also well-defined for other cases.

In this special case, let s_i be a repositioning flight in S. If $i = 1$, then we assume that s_1 departs as late as possible, i.e., $ed_f(s_1) = ld_f(s_1) = rd_f(s_2) - du_f(s_1)$. Otherwise, let us assume that s_i departs as soon as possible, i.e., set $ld_f(s_i) = ed_f(s_i) = ea_f(s_{i-1})$. It follows that each flight in S has a trivial time window in this special case, and attributes of duty S include the followings:

- $csi(S)$: crew start index.

$$csi(S) = \begin{cases} \text{argmin}_{1 \le i \le n}(s_i \text{ is not a trivial flight}), & \text{if } S \text{ is a crew-duty} \\ -1, & \text{otherwise.} \end{cases}$$

- $cei(S)$: crew end index.

$$cei(S) = \begin{cases} \text{argmax}_{1 \le i \le n}(s_i \text{ is not a trivial flight}), & \text{if } S \text{ is a crew-duty} \\ -1, & \text{otherwise.} \end{cases}$$

- $ft_f(S)$: flight time of S. $ft_f(S) = \sum_{i=1}^{m} ft_f(s_i)$.

- $du_f(S)$: duration of S. $du_f(S) = ea_f(s_m) - ed_f(s_1)$.

- $dt_f(S)$: duty time of S.

$$dt_f(S) = \begin{cases} ea_f(s_{cei(S)}) - tt_f(s_{cei(S)}) - ed_f(s_{csi(S)}), & \text{if } S \text{ is a crew-duty} \\ 0, & \text{otherwise.} \end{cases}$$

  Note that the turn time is subtracted from the arrival time.

With the above attributes of S defined, S is *feasible* duty of fleet type f if all the followings hold:

1. There is an airplane of fleet type f that is type-feasible for all demands in S.

2. $ft_f(S) \leq maxFlight$, in which $maxFlight$ is a given limit on the maximum flight time of a duty. $maxFlight$ is usually set to be 10 hours according to the regulations.

3. $dt_f(S) + preFlight + postFlight \leq maxDuty$, in which $maxDuty$ is a given limit on the maximum flight time of a duty, $preFlight$ is the time that crew need before starting the first flight of a duty, and $postFlight$ is the time that crew need after the last flight of a duty. $maxDuty$ is usually set to be 14 hours according to the regulations, and $preFlight$ and $postFlight$ are set to be 60 minutes.

   Note that if a duty is the first duty of a pilot's one-week tour, $preFlight$ is 90 minutes. And if a duty is the last duty of a pilot's one-week tour, $postFlight$ is 90 minutes. Therefore, when generating duties, we will set both $preFlight$ and $postFlight$ to be 60 minutes to check the feasibility of a duty, since we do not know whether this duty will be assigned as the first or last duty of the tour, or as a duty in the middle.

   When generating tours in the pricing subproblem, we set preFlight and post-Flight to be the correct values if applicable. In other words, a feasible duty may not be feasible in a tour, if it is the first or last duty of the tour, because of the maximum duty time constraint. But note that we do not skip any feasible duties by relaxing in this way.

An *airplane group* is defined to be a set of airplanes in $\mathbb{A}$, so that they are of the same fleet type and are *type-feasible* for the same set of demands. Given a fleet

type f, it follows that all airplane groups of type f form a partition of the set of all airplanes of type f. Therefore, all airplane groups form a partition of $\mathbb{A}$.

In the literature, a tour consists of crew's activities starting from the home base and ending at the home base. For fractional airlines, such a tour for a pilot usually spans one week, followed by a rest period of one week. However, we define a *tour* as part of the schedule of the current planning horizon that is related to a crew. In other words, it contains a pair of pilots, an airplane and a sequence of flights (including trivial flights) that are assigned to the crew and airplane in the current planning horizon.

Therefore, the factional airline scheduling problem (FASP) is to find a minimum cost set of tours such that all the followings hold: each tour must satisfy all feasibility rules, including crew-related rules; at any given time, each airplane must only be used by at most one tour, and any pilot must only be assigned to at most one tour; each demand can only be covered by all tours at most once.

2.2 Complexity

In this section, we will prove that all four special cases of the FASP are NP-complete.

Let us consider the following four attributes of the FASP:

1. There are maintenance and appointments;
2. Demands have time windows;
3. There are travel and crew-related constraints;
4. There are multiple fleet types.

For $1 \leq i \leq 4$, let $FASP^i$ be the special case of the FASP with the above attribute (i), but without the other three attributes. For example, $FASP^1$ is the FASP with maintenance and appointments and single aircraft type, but crews are not considered and demands have no time windows.

We will show that each of the special cases is NP-complete. Note that the case for $FASP^1$ follows directly from the results in Keskinocak and Tayur [16].

2.2.1 With Time Windows

In the Travelling Salesman Problem with time windows (TSPTW), there is travel time, t_{ij}, between each pair of vertices i and j, and there is a time window $[e_i, l_i]$ for each vertex i. The objective is to find a tour (containing all vertices) such that each vertex is served within its time window. More specifically, if the arrival time at vertex i is earlier than e_i, then the service must begin at e_i. If the arrival time at vertex i is later than l_i, then it is infeasible. And if the arrival time at vertex i is between e_i and l_i, then the service can start at the arrival time. Moreover, it is assumed that there is no service time. So the travelling to another vertex can begin at the service start time.

If we consider each vertex as a demand, then TSPTW is a special case of $FASP^2$ with a single airplane and no crew-related constraints. In Savelsbergh [25], it is proved that the feasibility problem for TSPTW is NP-complete. It follows that $FASP^2$ is NP-complete.

2.2.2 With Crews

Note that crew pairs are given as input and crew's travel is generated by an outside routine in our assumptions about $FASP$. We will prove the case of $FASP^3$ by using a reduction from the pairwise consistent 3-dimensional matching (3DM) (see Garey and Johnson [12]) which states the following problem:

Given a set $M \in W \times X \times Y$, where W, X and Y are disjoint sets with the same number q of elements. Moreover, M satisfies the following pairwise consistent property: for all elements $a \in W$, $b \in X$, $c \in Y$, whenever there exist elements $w \in W$, $x \in X$, $y \in Y$ such that $(a, b, y) \in M$, $(a, x, c) \in M$, and $(w, b, c) \in M$, then $(a, b, c) \in M$. The problem is to determine whether M contains a 3-dimensional

matching of size q.

Theorem 2.2.1. *$FASP^3$ is NP-complete.*

Proof. Given an instance of the 3DM, we can construct an instance of $FASP^3$ as follows: let W be the set of captain pilots; let X be the set of first officers; let Y be the set of demands; all the involved airports (available airport of each pilot and origin and destination of each demand) are distinct.

Moreover, the set of feasible crew pairs (a pair of pilots who can be assigned to the same duty) is induced from M. Namely, if there exists $(w, x, y) \in M$, then (w, x) is a feasible pair of pilots. For each $(w, x, y) \in M$, let a_w and a_x be the available airports of captain w and officer x respectively, and let a_y be the origin of demand y.

Furthermore, we will add all the following constraints to the $FASP^3$: all demands depart at the same time and have the same duration and cost; all pilots are available at the same time with enough duty time left to cover travel and demands; there is an airplane available for each demand at the same airport and the same time; it is feasible for captain w to travel from airport a_w to airport a_y and to arrive at a_y before the departure of demand y if and only if there exists $b \in X$ such that $(w, b, y) \in M$, and the same holds for the travel of any first officer $x \in X$; all feasible travels have the same duration and cost.

With the above constraints on the $FASP^3$, we can show that there is a one-to-one correspondence between a feasible schedule of $FASP^3$ without charter flights and a 3-dimensional matching of size q in M.

Given a 3-dimensional matching of size q in M, each "edge" (a, b, c) in the matching corresponds to a feasible tour in $FASP^3$, which means that pilots a and b travel to airport a_c to satisfy demand c. Moreover, this matching covers each captain, each officer and each demand exactly once, and has size q. Thus it corresponds to a feasible schedule of $FASP^3$ with each demand satisfied, i.e., no charter flights.

Similarly, given a feasible schedule of $FASP^3$ containing no charter flights, it is easy to check that it corresponds to a 3-dimensional matching of size q. We need to show that each tour in this schedule corresponds to an edge in M, thus this matching is contained in M.

Let (a, b, c) be an tour in the feasible schedule. Since it is feasible for captain a to travel from a_a to a_c, by the construction of feasible travel, we know that there exists $x \in X$ such that $(a, x, c) \in M$. Similarly, there exists $w \in W$ such that $(w, b, c) \in M$. Moreover, since (a, b, c) is a feasible tour, we know that (a, b) is a feasible pair of pilots, and by the construction of feasible crew pairs, there exists $y \in Y$ such that $(a, b, y) \in M$. Now it follows from the pairwise consistent property of M that $(a, b, c) \in M$. □

2.2.3 With Two Fleet Types

Consider $FASP^4$, and note that the costs for different types of airplanes to fly the same flight are different. We will show that $FASP^4$ is NP-complete by a reduction from the One-In-Three Satisfiability problem (One-In-Three 3SAT) with un-negated variables. The reduction in our proof is similar to the reduction for the multi-commodity matching problem in Bertossi et. al. [4].

Let U be the set of variables, and let C be the set of clauses over U such that each clause contains exactly three un-negated variables. The problem is to determine whether there exists a truth assignment for U such that each clause in C has exactly one true literal.

Theorem 2.2.2. *$FASP^4$ with two aircraft types is NP-complete.*

Proof. Given an instance of the One-In-Three 3SAT with n un-negated variables and m clauses, WLOG, we can order all clauses so that for each variable, the i-th appearance of this variable in the set of all clauses is well-defined. For each variable v_i, let $r(v_i)$ be the number of appearances of variable v_i in all clauses.

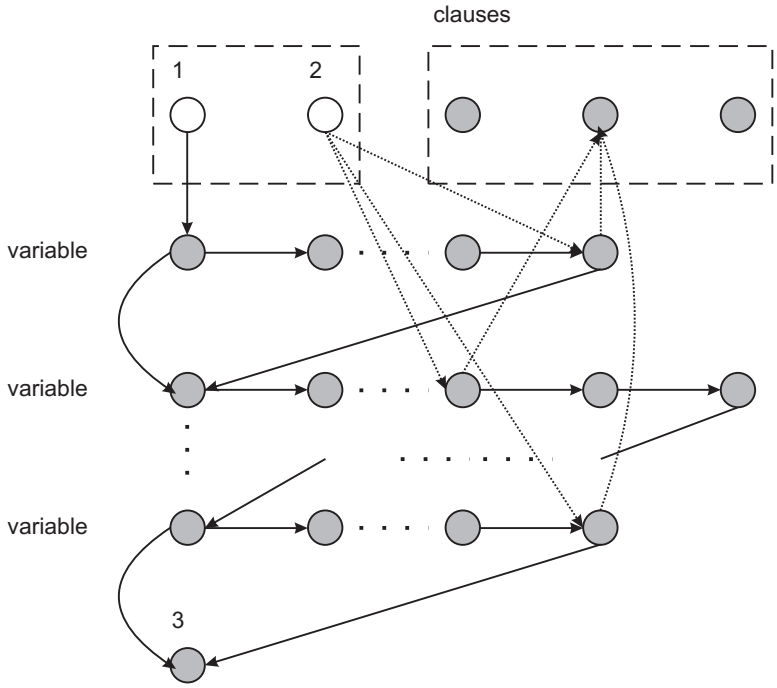


Figure 1: Reduction From 3-SAT

Based on the given 3SAT instance, let us construct a complete directed graph as in Figure 1 as the following:

- the first horizontal layer in the graph contains two special nodes s_1 and s_2 and nodes corresponding to all clauses: $(c_1, ..., c_m)$;
- the last horizontal layer consists of a single special node s_3;
- each of the other horizontal layers corresponds to a variable v_i.

More specifically, s_1 and s_2 correspond to two aircraft types. The layer corresponding to a variable v_i contains $r(v_i) + 1$ nodes $(n_0^i, n_1^i, ..., n_{r(v_i)}^i)$: the first node n_0^i is a special node representing the variable v_i; the j-th node n_j^i corresponds to the j-th appearance of variable v_i in the clauses.

There are three types of arcs in the graph. The first type of arcs contain the following arcs:

- arc $s_1 n_0^1$, $n_0^n s_3$ and $n_{r(v_n)}^n s_3$;

- for each variable v_i, arc $n_j^i n_{j+1}^i$ for $0 \leq j \leq r(v_i) - 1$;
- arc $n_0^i n_0^{i+1}$ and arc $n_{r(v_i)}^i n_0^{i+1}$ for $1 \leq i \leq n-1$.

The second type of arcs contain the following: for each variable v_i and $1 \leq j \leq r(v_i)$, arc $s_2 n_j^i$ and arc from n_j^i to the clause node c_k which has the j-th appearance of variable v_i. The third type of arcs consist of all other arcs.

Now we can convert the graph we constructed above to an instance of $FASP^4$ with two aircraft types.

Let s_1 be the base for type-1 airplanes and there is a single type 1 airplane. Similarly, let s_2 be the base for type-2 airplanes and there are m type-2 airplanes. All other nodes correspond to demands, and all airports are distinct. Each arc corresponds to a repositioning flight. Consider the sequence of nodes $P = (n_0^1, ..., n_{r(v_n)+1}^n)$, which consists of all nodes except for the first layer of nodes. We can set the departure and arrival time of each demand and repositioning flight in P such that it is feasible for any airplane to fly this sequence of flights. Moreover, we set all demands corresponding to clause nodes to start at the same time t_2 , and set all airplanes to be available at the same time t_1, so that all type-2 arcs are feasible.

It follows that each flight in P starts after t_1 and ends before t_2. So if a clause node is assigned to an airplane, it must be the last demand of this airplane. Moreover, the repositioning flight in the reverse direction of any type-1 or type-2 arc is not feasible, so we can delete these arcs from the graph.

Let $C = (3m + n + 1)(m + 1)$. We will define cost function $c_1(\cdot)$ on all arcs for type-1 airplanes and define cost function $c_2(\cdot)$ on all arcs for type-2 airplanes. For all type-1 arcs, we define the costs as the followings:

- $c_2(a) = C + 1$ for each type-1 arc a;
- $c_1(n_0^j n_0^{j+1}) = r(v_j) + 1$ for $1 \leq j \leq n-1$, and $c_1(n_0^n s_3) = r(v_n) + 1$;
- for each of other type-1 arcs a, $c_1(a) = 1$.

For all type-2 arcs, we define the cost functions as the following:

- $c_1(a) = C + 1$ for each type-2 arc a;
- $c_2(s_2 n_j^i) = 1$ for each type-2 arc $s_2 n_j^i$;
- $c_2(n_j^i c_k) = 3m + n$ for each type-2 arc $n_j^i c_k$.

For each type-3 arc a, $c_1(a) = c_2(a) = C + 1$. And we set the charter cost for each demand to be $C + 1$.

We will show that there is a one-to-one correspondence between a feasible schedule of the above constructed instance of $FASP^4$ with cost equal to C and a truth assignment for the given 3SAT instance such that each clause has exactly one true literal.

Suppose that S is a feasible schedule of $FASP^4$ with cost C. It follows that each demand is covered exactly once. Moreover, it follows from the definition of the cost function c_1 that clause nodes can only be assigned with type-2 airplanes. Since a clause node must be the last demand of a tour, each clause node is assigned with a unique type-2 airplane. Thus, the lower bound for a path from s_2 to a clause node is $3m + n + 1$.

Moreover, note that demand s_3 can only be assigned with a type-1 airplane in this schedule. In order to cover demand s_3, the type-1 airplane must be assigned to a path from s_1 to s_3. The lower bound for the cost of this path is $1 + \sum_{i=1}^{n}(r(v_i) + 1) = 3m + n + 1$. Therefore, the lower bound for the total cost of any feasible schedule without charter flights is $(3m + n + 1)(m + 1) = C$, which implies that S is optimal.

It also follows from the definition of the cost functions that, in S and for $1 \leq i \leq n$, the set of nodes $P^i = \{n_1^i, ..., n_{r(v_i)}^i\}$ is either assigned to a type-1 airplane or assigned to $r(v_i)$ different type-2 airplanes. In the first case, we set the variable v_i to be false, and in the second case, we set v_i to be true. It is easy to check that each clause c_k contains exactly one true literal, which is n_j^i such that $n_j^i c_k$ is in the schedule S.

Suppose that T is a truth assignment such that each clause has exact one true literal. We can construct a schedule as the following: if variable v_i is set to be true in T, then for $1 \leq j \leq r(v_i)$, each demand n_j^i is covered by the tour $s_2 - n_j^i - c_k$ and a unique type-2 airplane. The tour of the type-1 airplane is a path from s_1 to s_3 and covers all other nodes. Since each clause contains exactly one true literal, each clause node is covered by the tours exactly once. It follows that this schedule is feasible, contains no charter flights and has cost equal to C. □

CHAPTER III

NETWORK GENERATION AND OPTIMIZATION

In this chapter, we will discuss how to efficiently represent all feasible tours. More specifically, we will construct a network such that each feasible tour corresponds to a path in this network. Moreover, a path in this network must contain all the information that a tour represents. We will also demonstrate how to optimize and reduce the size of this network when we proceed with the constructing process.

3.1 Duty Generation

In the network that we want to construct, some nodes correspond to feasible duties. In this section, we will discuss properties of a duty with the minimum duration and minimum duty time and show how to generate them. In particular, we will define feasibility of a duty, construct a demand graph, and then find all feasible duties by traversing the demand graph. Then we will show how to time a duty in order to minimize its duration and total duty time, show how to keep all optimal timings of a duty by setting a time window for it (instead of just returning a single optimal timing), and define all attributes of a feasible duty with the minimum duration and duty time.

3.1.1 Demand Graph and Duty

For each fleet type f, a demand graph, $\mathbb{G}_f = (N, A)$, is a directed graph constructed as follows: each node $i \in N$ in $\mathbb{G}_f$ corresponds to a demand $i \in \mathbb{D}$, for which f is feasible; there is an arc from node i to node j if either of the followings holds:

- $j = ba(i)$, i.e., j is the demand bundled after i, or $j = bb(i)$, i.e., j is the demand bundled before i;

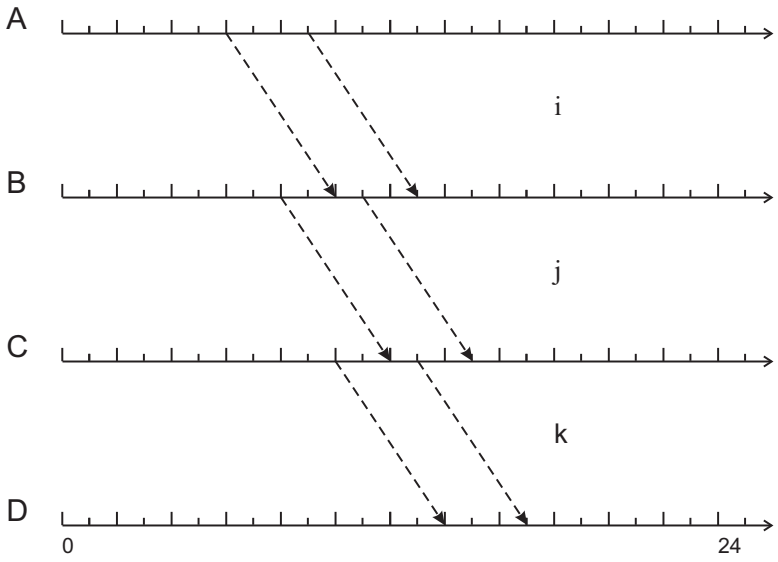


Figure 2: Path in demand graph

- no demand bundled after i or before j. Moreover, if $da(i) = oa(j)$, then $ea_f(i) \leq ld_f(j)$. Otherwise, the repositioning leg ij must be feasible.

Let us define an arc ij to be *trivial* arc if $da(i) = oa(j)$, i.e., no repositioning needed.

Demand graph is similar to the flight network in the case of commercial airlines. But our demand also includes maintenance, and another key difference is that, when there are time windows, a path in $\mathbb{G}_f$ may not be **feasible**. This is because that each arc in $\mathbb{G}_f$ is generated in a relaxed way. If there is enough time after the earliest arrival time of demand i and before the latest departure time of demand j, then there is an arc ij in $\mathbb{G}_f$. But demand i may arrive later, or demand j may depart earlier, resulting in infeasibility of the connection between them.

For example, in Figure 2, there are three demands i, j and k, which are from airport A to B, B to C, and C to D respectively. Elapsed time for each demand is equal to 4 hours. Time windows for demands i, j and k are [6:00, 9:00], [8:00, 11:00] and [10:00, 13:00] respectively. It follows that arc ij and jk are in the corresponding demands graph. However, the path (i, j, k) is not feasible because of the following:

if i precedes j, then demand j must arrive later than 14:00, which is later than the latest departure time of demand k.

Note that the maximum time window for any demand is six hours in practice, so we assume that the maximum time window is six hours.

Another property of a demand graph is that it may contain **cycles**. For example, in Figure 3, the demand k is from airport C to A, and it can be checked that arcs ij, jk and ki are generated.

Take a simple path $P = (n_0, a_0, n_1, a_1, \ldots, a_{m-1}, n_m)$ in $\mathbb{G}_f$, in which n_i is a node in $\mathbb{G}_f$, and a_i is the arc from node n_i to node n_{i+1} in $\mathbb{G}_f$. Delete all *trivial* arcs from P and denote the remaining set by $S = (s_0, s_1, \ldots, s_n)$. It follows that if $s_i \in S$, then s_i corresponds to a flight: a customer-requested flight, a trivial flight or a repositioning flight. S defines a duty *induced from* path P.

We will apply a depth-first-search algorithm, ALGORITHM 1, to the demand graph G_f to generate feasible duties. In particular, we will enumerate paths in G_f and generate duties induced by these paths. Note that after a path is enumerated, we need to check the time feasibility of this path, in addition to other feasibility rules of a duty.

When there is no confusion, we will use notation of demands and repositioning flight legs directly on nodes and arcs in S and drop the fleet type subscript f. For example, we will use $et(s_i)$, instead of et_f(the demand corresponding to s_i), for $s_i \in S$ to denote the elapsed time of the demand corresponding to s_i.

Definition 3.1.1. *A duty* $S = (s_0, s_1, \ldots, s_n)$ *is* time-feasible *if there exists a* n*-dimensional timing vector* $x = (x_1, x_2, \ldots, x_n)$ *such that* $ed(s_i) \leq x_i \leq ld(s_i)$ *for any* i, $1 \leq i \leq n$, *and* $x_i + et(s_i) \leq x_{i+1}$ *for any* i, $1 \leq i \leq n-1$.

In other words, S is time-feasible, if we can time S by setting the departure time of each flight to be an allowed departure time, so that all connections between nodes

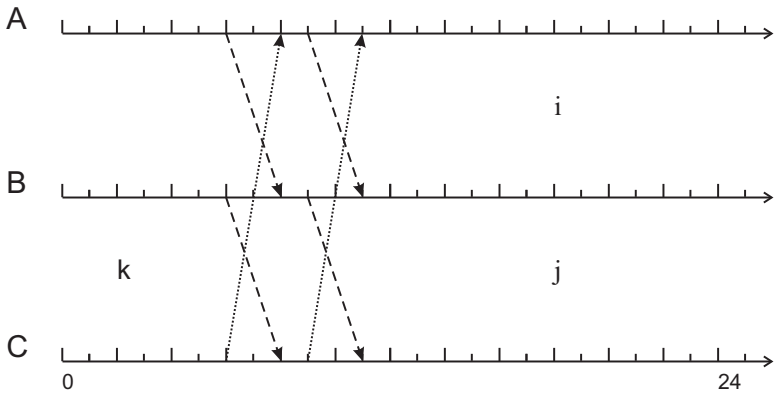


Figure 3: Cycle in demand graph

(demands) in S are feasible. Such a vector x is denoted as a feasible *timing* of duty S. Note that feasible timings of S may not be unique.

One key observation about a duty is the following: total flight time of a duty S does not depend on the timing of S, but total duty time and duration of S may depend on the timing.

For example, consider the duty S, induced by (i, j) in Figure 3, and two timings of S: T^1 = (6:00, 8:00) and T^2 = (6:00, 9:00). Duty times for these two timings are 4 and 5 hours respectively. Note that demand i departs at 6:00 in both timings. Therefore, if S is timed with T^2 in a schedule, it can always be re-timed with T^1 and the schedule is still feasible. And with T^1, S has the minimum duty time among all feasible timings of S. Moreover, we can check that the set $\{(x, x+2), \forall 6 \leq x \leq 7\}$ contains all feasible timings of S that induces a duty time of 4 hours. In other words, we can consider [6:00, 7:00] to be the time window for the start time of duty S and consider the duration of S to be 4 hours. Thus, if duty S starts at time $6+\Delta$, it ends at $6+\Delta+4$, and the corresponding optimal timing is (6:00 + Δ, 8:00 + Δ).

Suppose that no demands in S have time windows. Non-trivial repositioning legs in S are set to reposition as early as possible, except for the case when $csi(S) < cei(S)$ and $s_{csi(S)}$ is a repositioning leg. In this case, $s_{csi(S)}$ is set to reposition as late as possible. It follows that with the repositioning legs set this way, the total duty time

Algorithm 1 Depth First Search($\mathbb{G}_f$, P)

Require: demand graph $\mathbb{G}_f$, and path $P = (1, 2, \ldots, i) \in \mathbb{G}_f$.
Ensure: feasible duties.

```
for all j such that j ∉ P and ij ∈ G_f do
    if P ∪ j induces a feasible duty then
        populate duty induced from P ∪ j
        Depth First Search(G_f, P ∪ j)
return 0
```

of S is the minimum. Note that, in this case, attributes of a duty and feasibility of a duty have been defined in the previous section.

In general, duration and total duty time of a duty depend on how we time it. Let us define the time window for a duty as follows:

Definition 3.1.2. *$[ed(S), ld(S)]$ is a time window of duty S if the followings hold:*

1. *it is feasible for S to start at time $ed(S) + \delta$, for any $0 \leq \delta \leq ld(S) - ed(S)$;*
2. *if S starts at time $ed(S)+\delta$, then S can end at time $ed(S)+\delta+du(S)$, in which $du(S)$ is the minimum duration of duty S.*

In the next sections, we will formulate the problem of how to time a duty as an optimization problem, study properties of its optimal solutions, and show that time window, duration and total duty time of a duty are well defined for a time-feasible duty. In addition, we will propose algorithms to check whether a given duty is time-feasible and return its time window and other attributes if it is time-feasible.

3.1.2 Minimize Duration of A Duty

Given a duty $\mathbb{S} = (s_1, s_2, \ldots, s_n)$, consider the following problem:

$$\textit{Min} \quad x_n + et(x_n) - x_0 \tag{1}$$

$$\textit{s.t.} \quad ed(s_i) \leq x_i \leq ld(s_i), \qquad \forall\ 1 \leq i \leq n \tag{2}$$

$$x_i + et(s_i) \leq x_{i+1}, \qquad \forall\ 1 \leq i \leq n-1 \tag{3}$$

The optimization problem (1) is to find a feasible timing of S to minimize the duration of S. A lower bound for the objective function is $\sum_{i=1}^{n} et(s_i)$ when each inequality of type (3) is tight. Since $et(x_n)$ is a constant, we can remove it from the objective function.

The following proposition characterizes the set of all optimal timings.

Proposition 3.1.1. *Let $\mathbb{Q}$ be the set of optimal solutions to the optimization problem (1), and let $\mathbb{Q}^1 = \{x_1 : x \in \mathbb{Q}\}$. Exactly one of the followings holds:*

- *$\mathbb{Q} = \varnothing$ and problem (1) is infeasible;*
- *$|\mathbb{Q}| = 1$, i.e., one unique optimal solution;*
- *$|\mathbb{Q}| \geq 2$ and $|\mathbb{Q}^1| = 1$, i.e., departure time of the first flight is the same in all optimal solutions;*
- *$|\mathbb{Q}^1| \geq 2$, and there exist $\bar{x} \in \mathbb{Q}$ and $\Delta > 0$ such that all optimal solutions lie on a line segment defined by $x = \bar{x} + \delta U$, where $0 \leq \delta \leq \Delta$ and U is a n-dimensional all-one vector. Moreover, for any $x \in \mathbb{Q}$, all inequalities of type (3) hold with equality.*

Proof. Since the problem is bounded, there is at lest one optimal solution if the problem is feasible.

Suppose that $|\mathbb{Q}^1| \geq 2$. Note that $|\mathbb{Q}| \geq |\mathbb{Q}^1|$. Therefore, $|\mathbb{Q}| \geq 2$. Let $x^*, x^{**} \in \mathbb{Q}$. For $1 \leq i \leq n$, define the following indicator that indicates whether the ith pair of entries in x^* and x^{**} violates, i.e., whether their difference is equal to the difference of the first pair of entries in x^* and x^{**}. Let

$$\delta_i(x^*, x^{**}) = \begin{cases} 1, & x_i^* - x_i^{**} \neq x_1^* - x_1^{**}; \\ 0, & \text{otherwise.} \end{cases}$$

Let $\text{dif}(x^*, x^{**}) = \sum_{i=1}^{n} \delta_i(x^*, x^{**})$, i.e., number of violating pair of entries. We will show the following:

$$\text{If } |\mathbb{Q}^1| \geq 2,\ x^*, x^{**} \in \mathbb{Q} \text{ and } x_1^* - x_1^{**} > 0, \text{ then } \text{dif}(x^*, x^{**}) = 0. \tag{4}$$

In other words, we need to show that $x_i^* - x_i^{**} = x_1^* - x_1^{**}, \forall\, 1 \leq i \leq n$.

Let $T = \{(x^*, x^{**}) : x_1^* - x_1^{**} > 0, \text{dif}(x^*, x^{**}) \geq 1, x^*, x^{**} \in \mathbb{Q}\}$. If T is empty, then (4) holds. Otherwise, consider the following:

$$\text{Min dif}(x^*, x^{**}), \text{ s.t } (x^*, x^{**}) \in T \tag{5}$$

Note that $\text{dif}(x^*, x^{**}) \in \{0, 1, \ldots, n\}$. Therefore, the minimum in (5) is attained. Let x^* and x^{**} be optimal to (5). We will show contradictions derived from x^* and x^{**}, which implies that T is empty.

Let $\delta = x_1^* - x_1^{**}$, and let $m = \text{argmin}_{1 \leq i \leq n}(x_i^* - x_i^{**} \neq \delta)$, i.e., index of the first violating pair of entries. It follows that $m \geq 2$. Note that if $m = n$, then x^* and x^{**} have different objective values. Therefore, $2 \leq m \leq n - 1$. There are three cases as the following:

Case 1: $x_m^* - x_m^{**} > \delta$ and $\text{dif}(x^*, x^{**}) = 1$

Let $\gamma = x_m^* - x_m^{**} - \delta > 0$ and let us define $\bar{x^*}$ to be the following:

$$\bar{x^*}_i = \begin{cases} x_i^*, & \text{if } i \leq m - 1; \\ x_i^* - \min(\gamma, \delta), & \text{otherwise.} \end{cases}$$

We will show that $\bar{x^*}$ is feasible but has shorter duration, therefore contradicting the optimality of x^*.

Note that $ed(s_m) \leq x_m^{**} = x_m^* - \delta - \gamma \leq x_m^* - \min(\gamma, \delta) \leq x_m^* \leq ld(s_m)$. Therefore, $ed(s_m) \leq \bar{x^*}_m \leq ld(s_m)$. Moreover, $\text{dif}(x^*, x^{**}) = 1$ implies that $x_i^{**} = x_i^* - \delta$ for $i \neq m$. Therefore, if $i \geq m + 1$, then $ed(s_i) \leq x_i^{**} = x_i^* - \delta \leq x_i^* - \min(\gamma, \delta) \leq x_i^* \leq ld(S_i)$, thus $ed(s_i) \leq \bar{x^*}_i \leq ld(s_i)$. In addition, if $i \leq m - 1$, then $\bar{x^*}_i = x_i^*$. Therefore, $ed(s_i) \leq \bar{x^*}_i \leq ld(s_i)$, $\forall 1 \leq i \leq n$.

Furthermore, since $m \geq 2$ and $x^*_{m-1} - x^{**}_{m-1} = \delta$, we have that

$$x^*_{m-1} - \delta + et(s_{m-1}) = x^{**}_{m-1} + et(s_{m-1}) \leq x^{**}_m. \tag{6}$$

It follows that

$$\begin{aligned} \bar{x^*}_{m-1} + et(s_{m-1}) &= x^*_{m-1} + et(s_{m-1}) \\ &\leq x^{**}_m + \delta = x^*_m - \gamma \leq x^*_m - \min(\gamma, \delta) = \bar{x^*}_m. \end{aligned} \tag{7}$$

The first inequality in (7) follows from (6). Moreover, if $i \geq m+1$, then we have the following:

$$\bar{x^*}_{i-1} + et(s_{i-1}) = x^*_{i-1} + et(s_{i-1}) - \min(\gamma, \delta) \leq x^*_i - \min(\gamma, \delta) = \bar{x^*}_i \tag{8}$$

And if $i \leq m-2$, then $\bar{x^*}_i = x^*_i$ and $\bar{x^*}_{i+1} = x^*_{i+1}$. Therefore, $\bar{x^*}_i + et(s_i) \leq \bar{x^*}_{i+1}$, $\forall 1 \leq i \leq n-1$.

Therefore, $\bar{x^*}$ is feasible. However, $\bar{x^*}_n - \bar{x^*}_0 = x^*_n - \min(\gamma, \delta) - x^*_0 < x^*_n - x^*_0$, contradicting to the optimality of x^*.

Case 2: $x^*_m - x^{**}_m > \delta$ and $\text{dif}(x^*, x^{**}) \geq 2$

Let $\gamma = x^*_m - x^{**}_m - \delta > 0$ and let $\bar{x^*}$ be the following:

$$\bar{x^*}_i = \begin{cases} x^*_m - \gamma, & \text{if } i = m; \\ x^*_i, & \text{otherwise.} \end{cases}$$

Similar to **Case 1**, we can check that $\bar{x^*}$ is feasible. Moreover, $\text{dif}(x^*, x^{**}) \geq 2$ implies that $1 \leq m \leq n-2$. So we have $\bar{x^*}_n = x^*_n$ and $\bar{x^*}_0 = x^*_0$. Thus $\bar{x^*} \in Q$. Comparing $\bar{x^*}$ with x^{**}, we have that $\bar{x^*}_m - x^{**}_m = \delta$. However, $x^*_m - x^{**}_m > \delta$. Therefore, $\text{dif}(\bar{x^*}, x^{**}) = \text{dif}(x^*, x^{**}) - 1 \geq 1$, contradicting to how we choose x^* and x^{**}.

Case 3: $x^*_m - x^{**}_m < \delta$

Let $\xi = \delta - \max(x^*_m - x^{**}_m, 0) > 0$. Note that $\xi \leq \delta$. Let us define $\bar{x^{**}}$ to be the following:

$$\bar{x^{**}}_i = \begin{cases} x^{**}_i, & \text{if } i \geq m; \\ x^{**}_i + \xi, & \text{otherwise.} \end{cases}$$

For $i \leq m-1$, $ed(s_i) \leq x^{**}_i \leq x^{**}_i + \xi \leq x^{**}_i + \delta = x^*_i \leq ld(s_i)$, thus $ed(s_i) \leq \bar{x^{**}}_i \leq ld(s_i)$. Moreover, $\bar{x^{**}}_i = x^{**}_i$ for $i \geq m$. It follows that $ed(s_i) \leq \bar{x^{**}}_i \leq ld(s_i)$, $\forall 1 \leq i \leq n$.

Consider $\bar{x^{**}}_{m-1}$ and $\bar{x^{**}}_m$, and we have the following:

$$\begin{aligned} \bar{x^{**}}_{m-1} + et(s_{m-1}) &= x^{**}_{m-1} + \xi + et(s_{m-1}) \\ &= x^{**}_{m-1} + \delta - \max(x^*_m - x^{**}_m, 0) + et(s_{m-1}) \\ &= x^*_{m-1} + et(s_{m-1}) - \max(x^*_m - x^{**}_m, 0) \\ &\leq x^*_m - \max(x^*_m - x^{**}_m, 0) \\ &\leq x^{**}_m \end{aligned}$$

Moreover, for $2 \leq i \leq m-1$, the following holds:

$$\bar{x^{**}}_{i-1} + et(s_{i-1}) = x^{**}_{i-1} + et(s_{i-1}) + \xi \leq x^{**}_i + \xi = \bar{x^{**}}_i. \tag{9}$$

In addition, $\bar{x^{**}}_{i-1} = x^{**}_{i-1}$ and $\bar{x^{**}}_i = x^{**}_i$, $\forall m+1 \leq i \leq n$. Therefore, $\bar{x^{**}}_{i-1} + et(s_{i-1}) \leq \bar{x^{**}}_i$, $\forall 2 \leq i \leq n$. It follows that $\bar{x^{**}}$ is feasible. However, $\bar{x^{**}}_n - \bar{x^*}_0 = x^{**}_n - x^{**}_0 - \xi$, contradicting to the optimality of x^{**}.

Thus (4) is proved, and for any $x^*, x^{**} \in \mathbb{Q}$ such that $\delta = x^*_1 - x^{**}_1 \neq 0$, we have that $x^* = x^{**} + \delta U$. Let $x^a, x^b \in \mathbb{Q}$ such that $x^a_1 - x^b_1 = 0$. Since $|\mathbb{Q}^1| \geq 2$, there exists $x \in \mathbb{Q}$ such that $x_1 \neq x^a_1$. It follows that $x^a = x + (x^a_1 - x_1)U = x + (x^b_1 - x_1)U = x^b$.

Therefore, we proved that when $|\mathbb{Q}^1| \geq 2$, all solutions in $\mathbb{Q}$ lie on a line. Since $\mathbb{Q}$ is bounded, $\mathbb{Q}$ is a line segment that has two end points, defined by $\bar{x}$ and $\hat{x}$, and $\hat{x} = \bar{x} + \Delta U$, where $\Delta = \hat{x}_1 - \bar{x}_1$.

Now we will show the following: if $|\mathbb{Q}^1| \geq 2$ and $x \in \mathbb{Q}$, then all inequalities of type (3) hold with equality. Suppose that there exists $x \in \mathbb{Q}$ such that $x_i + et(s_i) < x_{i+1}$ for some index i, $1 \leq i \leq n$. It follows that $x = \bar{x} + \delta U$, in which $0 < \delta = x_1 - \bar{x}_1 \leq \Delta$. Let $m = \text{argmin}_{1 \leq i \leq n-1}(x_i + et(s_i) < x_{i+1})$, and let $\epsilon = \min(\Delta - \delta, x_{m+1} - x_m - ft(s_m)) > 0$, and let $x^* = x + \epsilon V$, in which

$$V = (\overbrace{1, 1, \ldots, 1}^{m}, 0, \ldots, 0)^T.$$

According to the definition of x^*, to show that x^* is feasible, we need to show the followings:

$$ed(s_i) \leq x_i^* \leq ld(s_i), \forall 1 \leq i \leq m, \tag{10}$$

and

$$x_m^* + et(s_m) \leq x_{m+1}^*. \tag{11}$$

For $1 \leq i \leq m$, note that $ed(s_i) \leq \bar{x}_i \leq \bar{x}_i + \delta + \epsilon \leq \bar{x}_i + \Delta = \hat{x} \leq ld(s_i)$ and $x_i^* = x_i + \epsilon = \bar{x}_i + \delta + \epsilon$. Therefore, (10) holds. Moreover, (11) is implies by $x_m + \epsilon + et(s_m) \leq x_{m+1}$ in the definition of ϵ. Therefore, x^* is feasible. However, $x_n^* - x_0^* = x_n - x_0 - \epsilon < x_n - x_0$, contradicting to the optimality of x. □

It follows from Proposition 3.1.1 that we have the following:

Theorem 3.1.1. *If $|\mathbb{Q}|^1 \geq 2$, let $\bar{x}$ and Δ be as defined in Proposition 3.1.1. If $|\mathbb{Q}|^1 = 1$, let $\bar{x} \in \mathbb{Q}$ and $\Delta = 0$. Then $du(S) = \bar{x}_n + et(s_n) - \bar{x}_1$, and the time window of S is $[ed(S), ld(S)] = [\bar{x}_1, \bar{x}_1 + \Delta]$.*

Note that $du(S)$ is defined as the minimum duration of S over all feasible timings of S, which is the optimal objective value of Problem (1). Therefore, duration of duty S is the same with any optimal timing of S.

Let us consider the case when $|\mathbb{Q}^1| \geq 2$. If $x \in \mathbb{Q}$, then $x_i = x_1 + \sum_{m=1}^{i-1} et(s_m), \forall 2 \leq i \leq n$. In other words, once x_1 is determined, all other entries in x are also determined. Thus there is a one-to-one correspondence between an optimal timing in $\mathbb{Q}$

Algorithm 2 Fix Leg Times($\mathbb{S}, i, t$)

Require: $\mathbb{S} = (s_1, s_2, \ldots, s_n)$, $1 \leq i \leq n$ and $ed(s_i) \leq t \leq ld(s_i)$
Ensure: $\mathbb{S}$ time-feasible, and optimal $x = \{x_i, \forall 1 \leq i \leq n\}$

$tmp \leftarrow t$, $x_i \leftarrow t$
for $k = i - 1$ to 1 **do**
    if $tmp - et(s_k) < ed(s_k)$ **then**
        return **false**
    else
        $x_k \leftarrow \min(ld(s_k), tmp - et(s_k))$
        $tmp \leftarrow x_k$
$tmp \leftarrow t$
for $k = i + 1$ to n **do**
    if $tmp + et(s_{k-1}) > ld(s_k)$ **then**
        return **false**
    else
        $x_k \leftarrow \max(ed(s_k), tmp + et(s_{k-1}))$
        $tmp \leftarrow x_k$
return **true**

and a start time in $[\bar{x}_1, \bar{x}_1 + \Delta]$, and the time window of S in this case represents the set of optimal solutions of Problem (1). Therefore, if duty S starts at time t, $\bar{x}_1 \leq t \leq \bar{x}_1 + \Delta$, then S is assumed to be timed with the unique optimal timing that corresponds to t, and duration of S with this timing is the minimum duration.

In the case when $|\mathbb{Q}^1| = 1$, let $\bar{x} \in \mathbb{Q}$. Then the time window of $\mathbb{S}$ is $[\bar{x}_1, \bar{x}_1]$, and the duration is $\bar{x}_n + et(s_n) - \bar{x}_1$. Note that, in this case, although both the time window and duration are fixed, there may be multiple optimal timings with the same start time. So $\bar{x}_1$ does not correspond to a unique optimal timing. This case will be discussed further in the section of minimizing total duty time of a duty.

3.1.3 Generate Duties with Minimum Duration

In order to solve the optimization problem (1), we propose ALGORITHM 2 and ALGORITHM 3 that will determine whether problem (1) is feasible, and return two extreme points if it is feasible. In the following, we will describe ALGORITHM 2 and ALGORITHM 3 and prove their correctness.

Given the duty consisting of a set of flights, $\mathbb{S} = \{s_1, s_2, \ldots, s_n\}$, ALGORITHM $2(\mathbb{S}, i, t)$ sets the departure time of flight s_i to be t, and then sequentially fix x_m, for m from $i-1$ to 1 and for m from $i+1$ to n, so that the duration of $\mathbb{S}$ is the minimum. Define Problem $P(i,t)$ be the Problem (1) with an additional constraint $x_i = t$. In the following theorem, we will show that ALGORITHM $2(\mathbb{S}, i, t)$ solves Problem $P(i,t)$.

Theorem 3.1.2. *Problem $P(i,t)$ is feasible if and only if* ALGORITHM $2(\mathbb{S}, i, t)$ *returns true. Moreover, if Problem $P(i,t)$ is feasible, then the solution returned by* ALGORITHM $2(\mathbb{S}, i, t)$ *is optimal and is an extreme point.*

Proof. If ALGORITHM $2(\mathbb{S}, i, t)$ returns true, then we can check that the solution it returns is feasible to Problem $P(i,t)$. Therefore, Problem $P(i,t)$ is feasible.

Suppose that Problem $P(i,t)$ is feasible. Let x^* be an optimal solution. Suppose that ALGORITHM 2 returns false at line 11 at the index k, $i < k \leq n$. Let x be the solution that the algorithm is constructing so far. It follows from line 11 that $x_{k-1} + et(s_{k-1}) > ld(s_k)$. However, $x^*_{k-1} + et(s_{k-1}) \leq x^*_k \leq ld(S_k)$. Therefore, $x_{k-1} > x^*_{k-1}$. Note that $x_i = x^*_i = t$ and $k - 1 \geq i$. Thus $k - 1 > i$. It follows that there exist j, $i \leq j < j + 1 \leq k - 1$ such that $x_j \leq x^*_j$ and $x_{j+1} > x^*_{j+1}$. However, we have that $x^*_{j+1} \geq max(x^*_j + et(s_j), ed(s_{j+1})) \geq max(x_j + et(s_j), ed(s_{j+1})) = x_{j+1}$, contradiction. Therefore, line (11) must return true. Similarly, line (4) must return true too. Therefore, ALGORITHM 2 must return a feasible solution.

Let x be the feasible solution returned by ALGORITHM $2(\mathbb{S}, i, t)$, and let $\tilde{x}$ be any feasible solution of Problem $P(i,t)$. We can prove the following:

$$x_m \leq \tilde{x}_m, \forall m > i, \text{ and } x_m \geq \tilde{x}_m, \forall m < i. \tag{12}$$

Let $m > i$. Suppose that $x_m > \tilde{x}_m$. Since $x_i = \tilde{x}_i = t$, there exist j, $i \leq j < j+1 \leq m$, so that $x_j \leq \tilde{x}_j$ and $x_{j+1} > \tilde{x}_{j+1}$. However, we have that $\tilde{x}_{j+1} \geq max(\tilde{x}_j + et(s_j), ed(s_{j+1})) \geq max(x_j + et(s_j), ed(s_{j+1})) = x_{j+1}$, contradiction. Therefore, $x_m \leq \tilde{x}_m$. Similarly, $m < i$ implies that $x_m \geq \tilde{x}_m$. Therefore, (12) holds. Note that

Algorithm 3 Get Duty Time Window($\mathbb{S}$)

Require: $S = (s_1, s_2, \ldots, s_n)$.
Ensure: $\mathbb{S}$ time-feasible, optimal extreme points x^* and x^{**}.

```
1: for i = 1 to n do
2:     if ed(s_i) = ld(s_i) or (i < n and ld(s_i) + et(s_i) ≤ ed(s_{i+1})) then
3:         if (Fix Leg Times(S, i, ld(s_i)) is true) then
4:             x* ← x
5:             x** ← x
6:             return true
7:         else
8:             return false
9: feasible ← false
10: for i = n to 1 do
11:     if (Fix Leg Times(S, i, ed(s_i)) is true) then
12:         x* ← x
13:         feasible ← true
14:         break;
15: if (feasible is true) then
16:     for i = 1 to n do
17:         if (Fix Leg Times(S, i, ld(s_i)) is true) then
18:             x** ← x
19:             feasible ← true
20:             break;
21: return feasible
```

(12) implies that the value of each coordinate in x is either a lower bound or an upper bound for the corresponding coordinate. Therefore, x can not be a convex combination of any two distinct feasible solutions, i.e., x is an extreme point.

Since the optimal solution x^* is also feasible, it follows from (12) that $x_n \leq x_n^*$ and $x_1 \geq x_1^*$, thus $x_n - x_1 \leq x_n^* - x_1^*$. Therefore, $x_n - x_1 = x_n^* - x_1^*$, and x, returned by Algorithm 2, is also be optimal to Problem $P(i, t)$. □

Let us consider Algorithm 3. Line 1 through 8 determines if there exists $1 \leq i \leq n$ such that either s_i has a trivial time window, i.e $ed(s_i) = ld(s_i)$, or $ld(s_i) + et(s_i) \leq ed(s_{i+1})$. In either of these two cases, Algorithm 2($\mathbb{S}, i, ld(s_i)$) is called to check time-feasibility of S, and find an optimal solution of Problem (1).

Line 9 through 14 is a *backward search*, for i from n to 1, to check if it is feasible

to set the departure time of s_i to be the earliest departure time $ed(s_i)$. The search stops if ALGORITHM 2 returns true. Line 15 through 20 is a *forward search*, for i from 1 to n, to check if it is feasible to set the departure time of s_i to be the latest departure time $ld(s_i)$. The search stops if ALGORITHM 2 returns true.

Theorem 3.1.3. *Problem (1) is feasible, if and only if* ALGORITHM 3 *returns true. Moreover, if Problem (1) is feasible, then the time window of* $\mathbb{S}$ *is* $[x_1^*, x_1^{**}]$*, and the duration of* $\mathbb{S}$ *is* $et(s_n) + x_n^* - x_1^*$*, in which* x^* *and* x^{**} *are outputs of* ALGORITHM 3*.*

Proof. If Algorithm 3 returns true, then a feasible solution is returned, therefore Problem (1) is feasible.

To show the other direction, suppose that Problem (1) is feasible. It follows that it has at least one optimal solution. Let $\mathbb{Q}$ be the set of all optimal solutions, and let $\mathbb{Q}^1 = \{x_1 : x \in \mathbb{Q}\}$.

If $|\mathbb{Q}^1| \geq 2$, it follows from Proposition 3.1.1 that $\mathbb{Q}$ is a line segment with two end points $\bar{x}$ and $\hat{x}$, and $\hat{x} = \bar{x} + (\hat{x}_1 - \bar{x}_1)U$, where $\hat{x}_1 - \bar{x}_1 > 0$ and U is the n-dimensional all-one vector. If $|\mathbb{Q}^1| = 1$ and $\bar{x} \in \mathbb{Q}$, then $x_1 = \bar{x}_1$ and $x_n = \bar{x}_n$ for any $x \in \mathbb{Q}$.

First we will show that line 1 through 8 solves Problem (1) in the case when $ed(s_i) = ld(s_i)$ or $ld(s_i) + et(s_i) \leq ed(s_{i+1})$. If $ed(s_i) = ld(s_i)$, then it follows from Theorem 3.1.2 that ALGORITHM 2$(\mathbb{S}, i, ld(s_i))$ returns an optimal solution.

Consider the case when $ld(s_i) + et(s_i) \leq ed(s_{i+1})$. Let $\bar{x}$ be an optimal solution of Problem (1). If $\bar{x}_i < ld(s_i)$, then define $\bar{x}^*$ to be the following:

$$\bar{x}_j^* = \begin{cases} ld(s_i), & \text{if } j = i; \\ \bar{x}_j, & \text{otherwise.} \end{cases}$$

It follows that $\bar{x}^*$ is still feasible. Moreover, note that $1 < i < n$. Therefore, $\bar{x}_n^* - \bar{x}_1^* = \bar{x}_n - \bar{x}_1$, and $\bar{x}^*$ is still optimal to Problem (1). If $\bar{x}_i = ld(s_i)$, then let $\bar{x}^* = \bar{x}$.

Since $\bar{x}_i^* = ld(s_i)$, $\bar{x}^*$ is also optimal to Problem $P(i, ld(s_i))$. It follows that Problem $P(i, ld(s_i))$ has optimal solutions, and any optimal solution of Problem $P(i, ld(s_i))$

is also optimal to Problem (1). By Theorem 3.1.2, ALGORITHM 2$(\mathbb{S}, i, ld(s_i))$ returned a solution that is optimal to Problem $P(i, ld(s_i))$, thus optimal to Problem (1). In general, we have the following:

If x is returned by ALGORITHM 2$(\mathbb{S}, i, ld(s_i))$ or ALGORITHM 2$(\mathbb{S}, i, ed(s_i))$,
for $1 \leq i \leq n$, then x is an extreme point of Problem (1). (13)

It follows from Theorem 3.1.2 that x is an extreme point of Problem $P(i, ld(s_i))$. Moreover, $x_i = ld(s_i)$ or $ed(s_i)$. Therefore (13) holds.

Now suppose that $\nexists\ i$ such that $ed(s_i) = ld(s_i)$ or $ld(s_i) + et(s_i) \leq ed(s_{i+1})$. We will show that the *backward search* in ALGORITHM 3 returns an optimal extreme point x^*, and the *forward search* in ALGORITHM 3 returns an optimal extreme point x^{**}. Moreover, we will show that $x^* = \bar{x}$ and $x^{**} = \hat{x}$ in the case when $|\mathbb{Q}^1| \geq 2$.

Let $\tilde{x} \in \mathbb{Q}$ such that $\tilde{x}_1$ is the minimum among all optimal solutions in $\mathbb{Q}$. There must exist m such that $1 \leq m \leq n$ and $\tilde{x}_m = ed(s_m)$. Otherwise, we have that $\tilde{x}_i > ed(s_i)$, $\forall\ 1 \leq i \leq n$, and it follows that there exists small enough $\epsilon > 0$ such that $\tilde{x} - \epsilon U$ is still feasible and optimal, but $\tilde{x}_1 - \epsilon < \tilde{x}_1$, contradiction. Note that if $|\mathbb{Q}^1| \geq 2$, then since all optimal solutions form a line segment, there is only one optimal solution in $\mathbb{Q}$ such that x_1 is minimum, thus $\tilde{x} = \bar{x}$.

Take m such that $1 \leq m \leq n$ and $\tilde{x}_m = ed(s_m)$, and let x^m be the solution returned by ALGORITHM 2$(\mathbb{S}, m, ed(s_m))$. Since $x^m_m = ed(s_m) = \tilde{x}_m$, it follows from Theorem 3.1.2 that x^m is also optimal. Note that if $|\mathbb{Q}^1| \geq 2$, then there is only one optimal solution in $\mathbb{Q}$ such that $x_m = ed(s_m)$, thus $x^m = \tilde{x}$.

Once Algorithm 2$(\mathbb{S}, i, ed(s_i))$ returns a feasible solution, the *backward search* (Line 9 through 14) in ALGORITHM 3 stops. Let us define k to be the following:

$$k = \text{argmax}_{1 \leq i \leq n}(\text{ALGORITHM 2}(\mathbb{S}, i, ed(s_i)) \text{ returns feasible}). \tag{14}$$

Let x^* be the solution returned by the *backward search*, and let x^k be the solution returned by ALGORITHM 2 $(\mathbb{S}, k, ed(s_k))$. Then $x^* = x^k$. We will show that x^k is an

optimal extreme point, and $x^k = x^m$ if $|\mathbb{Q}^1| \geq 2$.

It follows from the definition of k that $m \leq k$, and if $k = m$, then $x^k = x^m$. Assume that $m < k$. Then $x^m_m = ed(s_m) \leq x^k_m$. We can show that $x^m_1 \leq x^k_1$. Suppose that $x^m_1 > x^k_1$. Let us define x^t to be the following:

$$x^t_j = \begin{cases} x^m_j, & \text{if } 1 \leq j \leq m-1; \\ x^k_j, & \text{otherwise.} \end{cases}$$

It follows that x^t is feasible and $x^t_k = ed(s_k)$, thus x^t is feasible to Problem $P(k, s_k)$. However, $x^t_n - x^t_1 = x^k_n - x^m_1 < x^k_n - x^k_1$, contradicting to that x^k is optimal to Problem $P(k, s_k)$. Therefore, $x^m_1 \leq x^k_1$.

Similarly, we can show that $x^k_n \leq x^m_n$. Therefore, $x^k_n - x^k_1 \leq x^m_n - x^m_1$, which implies that $x^k_n - x^k_1 = x^m_n - x^m_1$, since x^m is an optimal solution of problem (1). Therefore, $x^k_1 = x^m_1$, $x^k_n = x^m_n$ and x^k is also an optimal solution. It follows from (13) that x^k is an extreme point of Problem (1). Note that if $|\mathbb{Q}^1| \geq 2$, then there is only one optimal solution in $\mathbb{Q}$ such that $x_1 = x^m_1 = x^k_1$, thus $x^k = x^m$. Therefore, $x^* = x^k = x^m = \tilde{x} = \bar{x}$.

Thus we showed that *backward search* returns an optimal extreme point of Problem (1), and it returns $\bar{x}$ when $|\mathbb{Q}^1| \geq 2$. Similarly, we have that if Problem (1) is feasible, then x^q, the solution returned by ALGORITHM $2(\mathbb{S}, q, ld(s_q))$ in which

$$q = \text{argmin}_{1 \leq i \leq n}(\text{ALGORITHM } 2(\mathbb{S}, i, ld(s_i)) \text{ returns feasible}),$$

is an optimal extreme point, and $x^q = \hat{x}$ if $|\mathbb{Q}^1| \geq 2$.

Note that *forward search* returns feasible if and only if *backward search* returns feasible (they may return the same solution if Problem (1) has a unique optimal solution). This shows that if the *backward search* returns infeasible, then there's no need to do the *forward search*. Therefore, line 15 is valid. □

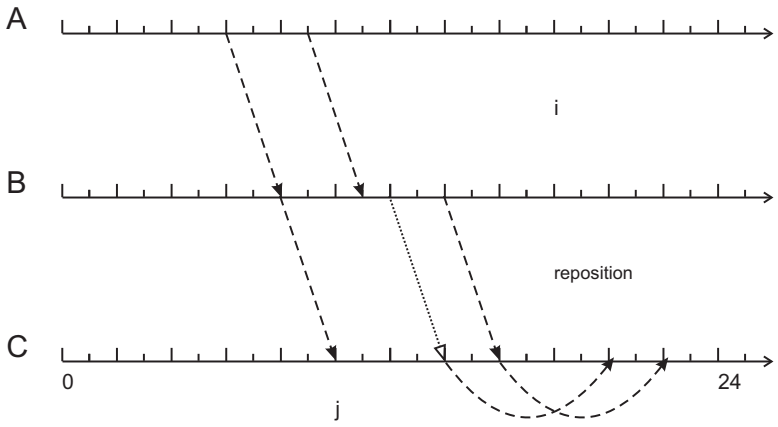


Figure 4: Duty time not minimized

3.1.4 Minimize Duty Time of A Duty

Let $S = (s_1, s_2, \ldots, s_n)$ be a duty induced from a path in $\mathbb{G}_f$. Some flights in S may be trivial flights that do no require crew. Therefore, the duty time of S may be less than $du(S)$. Moreover, note that the objective of Problem (1) is to minimize the duration, not duty time. Therefore, it is possible that duty time of S in an optimal timing of S (optimal with respect to Problem (1)) is not the minimum.

For example, consider Figure 4. i is a customer-requested flight from airport A to B, and j is a trivial flight corresponding to a maintenance request at airport C. There is a repositioning flight from B to C. Time window and duration of each flight are as denoted in the figure. These three flights correspond to an path (arc) (i, j) in the demand graph. Now consider the duty S induced by (i, j). The first iteration of *backward search* in ALGORITHM 3 will set $\bar{x}_j = ed(j) = 14{:}00$, then set the departure time of the repositioning flight, $\bar{x}_{ij}$, to be 12:00, as denoted by the dotted line in Figure 4, and then set $\bar{x}_i = ld(i) = 9{:}00$. The *search* the stops and returns $\bar{x}$. Duration of S is 11 hours, and duty time of S with timing $\bar{x}$ is 5 hours.

However, if we set $\bar{x}_{ij}$ to be 11:00 instead of 12:00, the resulting timing is still feasible, has the same duration, but has a duty time of 4 hours. Therefore, $\bar{x}$ does not give the minimum duty time in this example.

Observe that the time window in the example of Figure 4 is trivial, and the last flight is a trivial flight. We will show later that this is the only case when the duty time may not be the minimum in an optimal timing (optimal with respect to Problem (1)), and duty time can be minimized by applying ALGORITHM 3 one more time.

Assume that $S = (s_1, \ldots, s_n)$ is a feasible duty with the following properties:

$$ed(S) = ld(S), csi(S) < cei(S), \text{ and } csi(S) > 1 \text{ or } cei(S) < n. \tag{15}$$

Let $n^* = cei(S) - csi(S) + 1$ and let $S^* = (s_{csi(S)}, \ldots, s_{cei(S)}) \subset S$. In the following theorem, we will show that we can combine an optimal timing of S and an optimal timing of S^* to get an optimal timing of S that minimize both the duration and duty time of S.

Theorem 3.1.4. *Let $\bar{x}$ be an optimal timing of S, and let x^* be an optimal timing of S^*. Let $m = argmin_{csi(S) \le j \le cei(S)}(|\bar{x}_j - x^*_{j-csi(S)+1}|)$, and let us define δ to be the the followings:*

$$\delta = \begin{cases} 0, & \text{if } x^*_1 \ge \bar{x}_{csi(S)} \text{ and } x^*_{n^*} \le \bar{x}_{cei(S)} \ ; \\ \bar{x}_m - x^*_{m-csi(S)+1}, & \text{otherwise.} \end{cases}$$

For $1 \le j \le n$, define $\bar{x}^*_j$ to be the following:

$$\bar{x}^*_j = \begin{cases} x^*_{j-csi(S)+1} + \delta, & \text{if } csi(S) \le j \le cei(S); \\ \bar{x}_j, & \text{otherwise ;} \end{cases}$$

Then $\bar{x}^*$ is optimal to Problem (1) and has the minimum duty time over all feasible timings of S.

Proof. Let x^a be an optimal timing of S^*, and let x^b be a feasible timing of S^*. We will first prove the following:

$$\text{If } x^a_1 < x^b_1, \text{ then } x^a_j < x^b_j, \forall\ 1 \le j \le n^*. \text{ If } x^a_{n^*} > x^b_{n^*}, \text{ then } x^a_j > x^b_j, \forall\ 1 \le j \le n^*. \tag{16}$$

Assume that $x_1^a < x_1^b$. Since duration of S^* is the minimum with timing x^a, $x_{n^*}^a < x_{n^*}^b$. Suppose that there exists i such that $x_i^a \geq x_i^b$. It follows that $1 < i < n^*$. Let us define x^t to be the following:

$$x_j^t = \begin{cases} x_j^a, & \text{if } j \geq i; \\ x_j^b, & \text{otherwise.} \end{cases}$$

It follows that x^t is a feasible timing of S^*. However, $x_{n^*}^t - x_1^t = x_{n^*}^a - x_1^b < x_{n^*}^a - x_1^a$, contradicting to the optimality of x^a. Therefore, $x_j^a < x_j^b$, $\forall\ 1 \leq j \leq n^*$. Similarly, we can show that $x_{n^*}^a > x_{n^*}^a$. Thus (16) holds.

Now consider $\bar{x}$ and x^*, which are optimal timings of S and S^* respectively. Note that duration of S^* equals to the duty time of S, and x^* minimizes the duration of S^*. It follows that if $\bar{x}^*$, defined in Theorem 3.1.4, is a feasible timing of S, then $\bar{x}^*$ minimizes both the duty time and duration of S.

Let $x^{**} = x^* + \delta U = (\bar{x}_{csi(S)}^*, \ldots, \bar{x}_{cei(S)}^*)$, where δ is defined in Theorem 3.1.4. and U is a n^*-dimensional all-one vector. In other words, x^{**} is the restriction of $\bar{x}^*$ on S^*. In order to show that $\bar{x}^*$ is feasible, it suffices to show that x^{**} is a feasible timing of S^*, $x_1^{**} \geq \bar{x}_{csi(S)}$ and $x_{n^*}^{**} \leq \bar{x}_{cei(S)}$.

If $x_1^* \geq \bar{x}_{csi(S)}$ and $x_{n^*}^* \leq \bar{x}_{cei(S)}$, then $\delta = 0$ and $x^{**} = x^*$, which implies that x^{**} is a feasible timing of S^*. Otherwise, WLOG, assume that $x_1^* < \bar{x}_{csi(S)}$. Note that $(\bar{x}_{csi(S)}, \ldots, \bar{x}_{cei(S)})$ is a feasible timing of S^* and x^* is an optimal timing of S^*. It follows from (16) that $x_j^* < \bar{x}_{j-csi(S)+1}$, $\forall\ csi(S) \leq j \leq cei(S)$. Therefore, $\delta = min\{\bar{x}_j - x_{j-csi(S)+1}^* : csi(S) \leq j \leq cei(S)\} > 0$ and $x^{**} = x^* + \delta$ is also a feasible timing of S^*. Note that x^{**} is also an optimal timing of S^*, $(\bar{x}_{csi(S)}, \ldots, \bar{x}_{cei(S)})$ is a feasible timing of S^*, and $x_m^{**} = x_m^* + \delta = \bar{x}_m$. It follows from (16) that $x_1^{**} \geq \bar{x}_{csi(S)}$ and $x_{n^*}^{**} \leq \bar{x}_{cei(S)}$.

Similarly, if $x_{n^*}^* > \bar{x}_{cei(S)}$, we can show that $\delta = -min\{x_{j-csi(S)+1}^* - \bar{x}_j : csi(S) \leq j \leq cei(S)\} < 0$ and $\bar{x}^*$ is feasible. □

In the following, we will show that any timing in $\mathbb{Q}$ minimizes the duty time of S

in the cases other than (15):

1. $ed(S) < ld(S)$. If $csi(S) = cei(S) = -1$, then duty time of S is 0 with any timing of S. Otherwise, it follows from Proposition 3.1.1 that duty time of S with any timing in $\mathbb{Q}$ is $\sum_{j=csi(S)}^{cei(S)}(et(s_j))$, which is a lower bound for duty time of S with any feasible timing, thus minimum.

2. $csi(S) = cei(S)$. If $csi(S) = cei(S) = -1$, then duty time of S is 0. Otherwise, duty time of S is equal to $et(s_{csi(S)})$ in any feasible timing of S.

3. $csi(S) = 1$ and $cei(S) = n$. In this case, duty time of S is equal to duration of S. Therefore, any timing in $\mathbb{Q}$ also minimizes the duty time of S.

It follows from Theorem 3.1.4 and the above cases that $dt(S)$, the minimum duty time of S over all feasible timings of S, is well defined. Moreover, let us define $\tilde{Q}$ to be the set of feasible timings of S that have both minimum duration and minimum duty time. It follows that the followings hold:

- if time window of S is non-trivial, i.e., $ed(S) < ld(S)$, then any timing in $\mathbb{Q}$ that starts at a time in $[ed(S), ld(S)]$ minimizes both duration and duty time of S, thus $\tilde{Q} = \mathbb{Q}$;

- if time window of S is trivial, then any timing in $\mathbb{Q}$ starts at time $ed(S) = ld(S)$, and there exists a timing in $\mathbb{Q}$ that minimizes duration and duty time of S, thus $\varnothing \neq \tilde{Q} \subseteq \mathbb{Q}$.

3.1.5 Generate Attributes of A Duty

In the case when demands in a duty S have non-trivial time windows, we have shown so far that attributes of S such as $du(S)$, $[ed(S), ld(S)]$, $dt(S)$ are well-defined. Moreover, if S is feasible, then we know that there must exist feasible timing for S such that both duration and duty time of S are minimized, i.e., $\tilde{Q}(\subseteq \mathbb{Q}) \neq \varnothing$.

In the following, we will define and generate time window for the crew start time of a duty, and show that it corresponds to a set of timings that minimize both duration and duty time.

Let S be a crew-duty. As considered in the previous section, two different timings of S that have the same duration, duty time, start and end time may have different crew start and crew end times. Similar to the time window for the start time of duty S, we define the time window for the crew start time of duty S to be $[ecs(S), lcs(S)]$, where $ecs(S)$ is the *early crew start time* of S and $lcs(S)$ is the *late crew start time* of S.

Definition 3.1.3. *$[ecs(S), lcs(S)]$ is a time window for the crew start time of duty S if the followings hold: it is feasible for crew to start S at time $ecs(S) + \delta$, in which $0 \leq \delta \leq lcs(S) - ecs(S)$; if crew start at time $ecs(S) + \delta$, then they end at time $ecs(S) + \delta + dt(S)$.*

Note that $ed(s_{csi(S)}) \leq ecs(S) \leq ld(s_{csi(S)})$. Let $ece(S) = ecs(S) + dt(S)$ be *early crew end time* of duty S and let $lce(S) = lcs(S) + dt(S)$ be *late crew end time* of duty S.

In the following, we will find a set of optimal timings $\bar{Q} \subseteq \tilde{Q}$ such that there is a one-to-one correspondence between a time t, $ecs(S) \leq t \leq lcs(S)$, and an optimal timing x in $\bar{Q}$ with $x_{csi(S)} = t$.

Assume $csi(S) \geq 0$ and assume that ALGORITHM 3 has been applied to S, and x^* and x^{**} are returned. $ecs(S)$ and $lcs(S)$ can be generated as follows:

case 1 : $ed(S) < ld(S)$, or $csi(S) = 1$ and $cei(S) = n$. Then we have the following:

$$\begin{aligned} &ecs(S) = x^*_{csi(S)},\ lcs(S) = x^{**}_{csi(S)}, \\ &ece(S) = x^*_{cei(S)} + et(s_{cei(S)}) \text{ and } ece(S) = x^{**}_{cei(S)} + et(s_{cei(S)}). \end{aligned} \tag{17}$$

Let us set $\bar{Q} = Q$.

case 2 : $ed(S) = ld(S)$, and $csi(S) > 1$ or $cei(S) < n$. WLOG, let us assume that $csi(S) > 1$ and $cei(S) < n$. We can re-write S as $S = (s_1, \ldots, s_{csi(S)-1}) \cup (s_{csi(S)}, \ldots, s_{cei(S)}) \cup (s_{cei(S)+1}, \ldots, s_n) = S^1 \cup S^* \cup S^2$. Since S is feasible, S^1, S^* and S^2 are all feasible. Let us proceed with the following steps:

1. minimize the duration of S^1 with departure time of s_1 fixed to be $ed(S) = ld(S)$, and let x^1 be returned by ALGORITHM 2$(S^1, 1, ed(S))$;
2. minimize the duration of S^2 with departure time of s_n fixed to be $ee(S) = le(S)$, and let x^2 be returned by ALGORITHM 2$(S^2, n, ee(S))$;
3. apply ALGORITHM 3 to S^*, and let x^* and x^{**} be the returned optimal timings.

It follows from Theorem 3.1.4 that $du(S^*) = dt(S)$, and there exists t such that $ed(S^*) \leq t \leq ld(S^*)$, $t \geq x^1(s_{csi(S)-1}) + et(s_{csi(S)-1})$ and $t + du(S^*) \leq x^2(s_{cei(S)+1})$. Therefore, the followings are well-defined:

$$ecs(S) = max(x^1(s_{csi(S)-1}) + et(s_{csi(S)-1}), ed(S^*)),\ ece(S) = ecs(S) + du(S^*),$$
$$lcs(S) = min(ld(S^*), x^2(s_{cei(S)+1}) - du(S^*)) \text{ and } lce(S) = lcs(S) + du(S^*) \tag{18}$$

Let us set $\bar{Q} = (x^1, x^* + (t - ed(S^*))U, x^2)$, where $ecs(S) \leq t \leq lcs(S)$ and U is the n^*-dimensional all-one vector.

Moreover, in in (18), $ecs(S) = ed(S^*)$ in the case when $csi(S) = 1$ and $cei(S) < n$, and $lcs(S) = ld(S^*)$ in the case when $csi(S) > 1$ and $cei(S) = n$.

It follows that, in both case 1 and 2 above, $ecs(s)$ and $lcs(S)$ are well-defined, $\bar{Q} \subseteq \tilde{Q}$, and there is a one-to-one correspondence between a time t in $[ecs(S), lcs(S)]$ and an optimal timing x in $\bar{Q}$ with $x_{csi(S)} = t$.

To summarize, given a duty $S = (s_1, \ldots, s_n)$, we can use the following two-step process to determine whether S is time-feasible, and find time window, duration, crew

time window and duty time of S.

1. Apply ALGORITHM 3 to S. If it returns false, then duty S is infeasible and stop. Otherwise, ALGORITHM 3 returns x^* and x^{**}, and we have that $ed(S) = x_1^*$, $ld(S) = x_1^{**}$ and $du(S) = x_n^* + et(s_n) - x_1^*$.

2. If $ed(S) = ld(S)$, and $csi(S) > 1$ or $cei(S) < n$, then re-write S as $S^1 \cup S^* \cup S^2$. WLOG, let us assume that $csi(S) > 1$ and $cei(S) < n$. Thus both S^1 and S^2 are not empty. Let x^1 be returned by applying ALGORITHM $2(S^1, 1, x_1^*)$ to S^1, let x^2 be returned by applying ALGORITHM $2(S^1, n, x_n^*)$ to S^2, and let x^3 and x^4 be two extreme points returned by ALGORITHM 3 on S^*.

After the above two steps, if S is a crew-duty, then crew time window $[ecs(S), lcs(S)]$ can be obtained as in (17) and (18). Two end points of $\bar{Q}$ are the followings:

$$x^* = \begin{cases} x^*, & \text{if } csi(S) = -1 \text{ or } ed(S) < ld(S) \\ (x^1, x^3 + (ecs(S) - ed(S^*))U, x^2), & \text{otherwise.} \end{cases}$$

and

$$x^{**} = \begin{cases} x^{**}, & \text{if } csi(S) = -1 \text{ or } ed(S) < ld(S) \\ (x^1, x^3 + (lcs(S) - ed(S^*))U, x^2), & \text{otherwise.} \end{cases}$$

And the duty time of S is the following:

$$dt(S) = \begin{cases} 0, & \text{if } csi(S) = cei(S) = -1; \\ du(S^*) = x^*_{cei(S)} + et(s_{cei(S)}) - x^*_{csi(S)}, & \text{otherwise.} \end{cases}$$

3.2 Duty Network

In this section, we will discuss how to generate duty networks, properties of duty networks, dominance of paths in a duty network and aggregation of arcs in a duty network. Note that a path in a duty network that corresponds to a tour also contains crew-travelling information and how crew picks up the airplane that is assigned to

the tour. However, we will discuss them separately in the later sections because of their importance.

Let $S = (s_1, s_2, \ldots, s_n)$, $n \geq 1$, be a feasible duty with attributes defined as in the previous section. Note that s_1 and s_n are demands. So S may not capture crew's entire duty between rests. For example, it is possible that there is a non-trivial repositioning leg from the end airport of s_n to another airport, and crew will rest after this leg. Similarly, there may be a non-trivial repositioning leg right before s_1. If we want to enumerate all feasible duties, we need to consider all possible combinations of repositioning leg before s_1, S and repositioning leg after s_n.

In other words, let $m \geq 3$ and $P = (n_1, n_2, \ldots, n_m)$ be a simple path in G_f, and let a_i be the arc from n_i to n_{i+1} for $1 \leq i \leq m-1$. Let us assume that both a_0 and a_{m-1} are non-trivial, i.e., there is repositioning. We have considered the duty induced by $S = (n_1, a_1, n_2, \ldots, a_{m-1}, n_m)$, but we also need to consider the duties induced by $(a_1, n_2, \ldots, n_{m-1}, a_{m-1})$, $(a_1, n_2, \ldots, n_{m-1})$ and $(n_2, \ldots, n_{m-1}, a_{m-1})$.

The key observation is that if we enumerate all these duties, then the number of total duties will explode. Therefore, we only enumerate duties starting and ending with a demand, which corresponds to simple paths in the demand graph G_f. And we set the arc between duties to contain the possible repositioning leg, instead of also enumerating duties starting or ending with a repositioning leg.

3.2.1 Path

Given a fleet type f, the duty network of fleet type f is a directed graph $\mathbb{N}_f = (N, A)$. The set of nodes is $N = n_s \cup N_{dy} \cup N_{da} \cup N_{db}$. n_s is the unique source node. N_{dy} is the set of duty nodes, and each duty node corresponds to a feasible duty of fleet type f. N_{da} and N_{db} are sets of demand nodes. Each demand node corresponds to a demand that is feasible for fleet type f.

The set of arcs is $A = A_p \cup A_{dy} \cup A_d$, where A_p consists of all *pickup* arcs that are

from the source node n_s to a duty node or demand node, A_{dy} consists of arcs between duty nodes, and A_d consists of arcs from a duty node to a demand node.

When there is no confusion, we will use the term "duty node" and "duty" interchangeably and apply notation of a duty to a duty node directly. For example, if $n \in N_{dy}$, then $ft(n)$ refers to the total flight time of the duty that the duty node n corresponds to. Similarly, we will use the notation of demands for demand nodes.

Recall that we want to construct duty network N_f so that any feasible tour corresponds to a path in N_f that starts from the source node n_s and ends at a node in $N_{db} \cup N_{da}$.

Moreover, let P be a path in N_f and let S be the tour corresponding to S. If P ends at a demand node $d \in N_{db}$, then the crew of tour S will drop off the airplane of tour S before demand d, i.e., the airplane is dropped off at the airport $oa(d)$ after tour S, so that it can be picked up by another crew to perform a duty starting with demand d. Therefore, there is a repositioning flight from the last demand of tour S to demand d.

Similarly, if path P ends at a demand node $d \in N_{da}$, then the crew of tour S will drop off the airplane after demand d, i.e., d is the last demand of tour S and the airplane is dropped off at the airport $da(d)$ after tour S, so that it is ready to be picked up by another crew.

A pickup arc is associated with a pair of pilots and an airplane. Let $b \in A_p$ be a pickup arc that ends at a duty node $n \in N_{dy}$. Pickup arc b implies that it is feasible to assign its associated crew and airplane to perform duty n. If $b \in A_p$ is a pickup arc that ends at a demand node $d \in N_{db}$, then pickup arc b implies a feasible tour for the associated crew that consists of a single flight repositioning the associated airplane to $oa(d)$. Note that it follows from the definition of demand nodes that there is no pickup arc to a demand node in N_{da}.

Pickup arcs also involve crew travellings. We will discuss properties of travel and

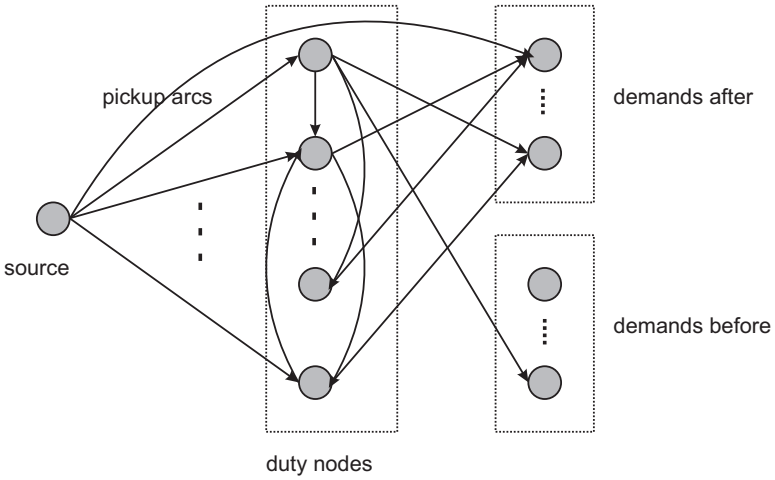


Figure 5: An example of duty network

pickup arcs, and how to generate them in the following sections.

3.2.2 Arcs Between Duty Nodes

Let us first consider the arcs between duty nodes in N_f. We will define the feasibility of such an arc, and then show that not all feasible arcs need to be present in N_f, after considering dominance of duties and paths in N_f. Arc aggregation not only reduces the size of the network, but also ensures that the N_f contains no cycles under realistic assumptions about the time windows.

3.2.2.1 Definition

Let $n_1, n_2 \in N_{dy}$ be two distinct duties, and let s be the repositioning flight from n_1 to n_2. If s is a trivial flight, then only a single no-repo arc from n_1 to n_2 is needed. Otherwise, there are at most three cases as in the following:

1. Repo-early. Repositioning flight s is combined with duty n_1 to form a duty $n_1 \cup s$, and crew rest after $n_1 \cup s$ and before n_2. It follows that s departs as early as possible after n_1.

2. Repo-late. Repositioning flight s is combined with duty n_2 to form a duty $s \cup n_2$, and crew rest after n_1 and before $s \cup n_2$. s departs as late as possible after n_1

in this case.

3. Repo-only. Repositioning flight s forms a duty by itself, and crew rest after n_1 and before s, and also rest after s and before n_2. So in this case, there is a duty between n_1 and n_2, and this duty consists of only a repositioning flight.

Note that all the above three cases may be necessary, because they represent three different tours $(n_1 \cup s, n_2)$, $(n_1, s \cup n_2)$ and (n_1, s, n_2).

Definition 3.2.1. *A no-repo arc from n_1 to n_2 is feasible if all the followings hold:*

- *the repositioning flight from n_1 to n_2 is trivial;*
- $ee(n_1) \leq ld(n_2)$*;*
- *if $csi(n_1) \geq 1$ and $csi(n_2) \geq 1$, then $ece(n_1) + minRest \leq lcs(n_2)$.*

Definition 3.2.2. *Let s be a nontrivial repositioning flight from n_1 to n_2.*

1. *A* repo-early *arc from n_1 to n_2 is feasible if all the followings hold:*
   - *$n_1 \cup s$ forms a feasible duty S;*
   - $ee(S) \leq ld(n_2)$*;*
   - *if n_2 is a crew-duty, then $ece(S) + minRest \leq lcs(n_2)$.*
2. *A* repo-late *arc from n_1 to n_2 is feasible if all the followings hold:*
   - *$s \cup n_2$ forms a feasible duty S;*
   - $ee(n_1) \leq ld(S)$*;*
   - *if n_1 is a crew-duty, then $ece(n_1) + minRest \leq lcs(SL)$.*
3. *Let*

$$t_1 = \begin{cases} \max(ece(n_1) + minRest, ee(n_1)), & \text{if } n_1 \text{ is an crew-duty} \\ ee(n_1), & \text{otherwise.} \end{cases}$$

and let

$$t_2 = \begin{cases} \min(lcs(n_2) - minRest, ld(n_2)), & \textit{if } n_2 \textit{ is an crew-duty} \\ ld(n_2), & \textit{otherwise.} \end{cases}$$

Then a repo-only *arc from n_1 to n_2 is feasible if the repositioning flight s can depart after time t_1 and arrive before time t_2.*

It follows that A_{dy}, the set of all between-duty arcs in the duty network N_f, only need to contain feasible arcs defined as above. Since an arc in A_{dy} corresponds to a repositioning flight, when there is no confusion, we will apply the notation of flight legs to the arcs directly.

3.2.2.2 Path Dominance

In this section, we will show sufficient conditions for a path to dominates another path in N_f. Note that we only consider paths consisting of duty nodes, i.e., paths starting at a duty node and ending at a duty node. This assumption holds throughout this section. Results in this section also constitute the bases for the arc aggregation in the next section.

Let us consider duties induced by a duty node and its incident arc(s). Suppose that $n \in N_{dy}$ is a duty node, $a \in A_{dy}$ is an arc into node n and $b \in A_{dy}$ is an arc out of n. Then $n \cup b$ induces a duty as follows: if b is an repo-early arc, then this duty consists of n and the early repositioning leg that b corresponds to; otherwise, this duty is n. Let us denote this induced duty by $S(n, b)$. Similarly, $S(a, n)$ is the duty induced by $a \cup n$, and $S(a, n, b)$ is the duty induced by $a \cup n \cup b$. Note that $S(a, n, b)$ contains n.

When generating an arc in A_{dy}, we have checked the feasibility of a duty induced by this arc and its incident duty node. Therefore, we know that both $S(a, n)$ and $S(n, b)$ are feasible duties, but $S(a, n, b)$ may not be feasible. Note that the results

in the previous section also hold for duties induced by a duty node and its incident arc(s).

In the following, we will define the dominance between two paths in N_f. Let $H = (n_1, a_1, n_2, \ldots, n_m)$ be a path in $\mathbb{N}_f$. Since there may be more than one arc between two duties, there may exist a time-feasible path $\hat{H} = (n_1, \hat{a}_1, n_2, \ldots, n_m)$ that contains the same set of nodes as H, but containing a different set of arcs. Therefore H and $\hat{H}$ may have different characteristics, and represent two different tours.

Note that departure time and arrival time of each flight in any feasible schedule are fixed. In other words, a schedule also implies a timing of all flights in this schedule.

Definition 3.2.3. *Let U be any feasible schedule that contains path $H \subseteq N_f$. After replacing path H with path $\hat{H} \subseteq N_f$ and keeping the timing of $U \backslash H$ to be the same, if schedule U is still feasible and the total cost of U does not increase, then $\hat{H}$* dominates *H.*

If we replace H with $\hat{H}$, then the new schedule covers the same set of demands and uses the same set of crews and airplanes. Therefore, if $\hat{H}$ *dominates* H, then we can always substitute H with $\hat{H}$ in any schedule containing H. It follows that H is redundant and we only need to consider $\hat{H}$ when searching for an optimal schedule.

Definition 3.2.4. *Let H and $\hat{H}$ be two feasible paths in $\mathbb{N}_f$. Then $\hat{H}$ is equivalent to H if $\hat{H}$ dominates H and H dominate $\hat{H}$.*

Let H be a path in N_f. We can write H as lists of different elements as the followings:

$$H = (n_1, a_1, n_2, \ldots, n_m) = (S_1, \ldots, S_k) = (s_1, \ldots, s_q) = (h_1, \ldots, h_p)$$

where

- n_i is a duty node and a_i is an arc, for $1 \leq i \leq m$;

- S_i is the i-th duty in H, for $1 \leq i \leq k$, e.g., $S_1 = S(n_1, a_1)$ and $S_k = S(a_{m-1}, n_m)$;
- s_i is the i-th flight in H, for $1 \leq i \leq q$;
- h_i is the i-th element in H, which is a duty or arc, for $1 \leq i \leq p$.

Moreover, let $h_{csi(H)}$ be the first element in $H = (h_1, \ldots, h_p)$ that requires crew, and let $h_{cei(H)}$ be the last element in H that requires crew.

For each duty $S_i \in H = (S_1, \ldots, S_k)$, time window and duration of S_i are well-defined. Moreover, after adding required minimum rest time (denoted by $minRest$) to the duration of each S_i, except for S_k, we can consider each duty S_i as a "flight" in the previous section that has time window $[ed(S_i), ld(S_i)]$ and duration $du(S_i)+minRest$, and the results in the previous section hold for $H = (S_1, \ldots, S_k)$. Therefore, time window and duration of path H are also well defined, and we will use the same notation for path H, i.e., $ed(H)$, $ld(H)$ and $du(H)$.

Furthermore, let us consider the case when $ed(H) = ld(H)$, and either s_1 or s_q is a trivial flight. Similar to the case when H is a duty, we need to consider crew start and end time of H. WLOG, assume that both s_1 and s_q are trivial flights, and both s_2 and s_{q-1} are nontrivial flights. Let $H^* = (S_1 \backslash s_1, S_2, \ldots, S_{k-1}, S_k \backslash s_q)$ and let $dt(H) = du(H^*)$. Note that $dt(H)$ is not the actual total crew duty time of H, because crew rest between duties in H. In this case, $dt(H)$ simply means that if crew start H at time t, then crew finish H at time $t + dt(H)$. It also follows that $ecs(H)$, $lcs(H)$ and $\bar{Q}_H$ are well defined.

The following theorem states a sufficient condition for path $\hat{H}$ to dominate path H.

Theorem 3.2.1. *Let $H = (n_1, a_1, n_2, \ldots, n_m) = (S_1, \ldots, S_k) = (s_1, \ldots, s_q)$ and $\hat{H} = (n_1, \hat{a}_1, n_2, \ldots, n_m) = (\hat{S}_1, \ldots, \hat{S}_r) = (s_1, \ldots, s_q)$ be two distinct feasible paths in $\mathbb{N}_f$*

that have the same set of nodes. Let $m \geq 2$. Then $\hat{H}$ dominates H if all the following conditions hold:

(a) $dt(\hat{H}) \leq dt(H)$;

(b) $lcs(\hat{H}) \geq lcs(H)$ *and* $ece(\hat{H}) \leq ece(H)$;

(c) $ft(\hat{S}_1) \leq ft(S_1)$ *and* $ft(\hat{S}_r) \leq ft(S_k))$;

Proof. Let $U = (H^1, a_0, H, a_m, H^2)$ be a schedule that is assigned to an airplane and crew. In U, H^1 and H^2 are paths in $\mathbb{N}_f$, a_0 is the arc from H^1 to H, and a_m is the arc from H to H^2. Let x be any feasible timing of U.

Let $\hat{U} = (H^1, a_0, \hat{H}, a_m, H^2)$. In order to show that $\hat{H}$ dominates H, we need to show that, when the timing of $\hat{U} \backslash \hat{H}$ is fixed to be the same as in x, $\hat{U}$ is still feasible. We will construct y, a timing of $\hat{U}$, so that: $y(s_i) = x(s_i)$ if $s_i \in \hat{U} \backslash \hat{H}$; y is time-feasible; each duty in $\hat{U}$ is feasible when timed with y.

WLOG, let us assume that both $\hat{S}_1$ and $\hat{S}_r$ are crew duties. Condition (c) implies that if $\hat{a}_1$ is a repo-early arc, then a_1 must also be a repo-early arc, thus $\hat{S}_1 = S(n_1, \hat{a}_1) \subseteq S(n_1, a_1) = S_1$. Similarly, $\hat{S}_r \subseteq S_k$. Thus, $dt(\hat{S}_1) \leq dt(S_1)$ and $dt(\hat{S}_r) \leq dt(S_k)$.

Let $h = csi(H)$. Then $h = csi(H) = csi(S_1) = csi(\hat{H}) = csi(\hat{S}_1)$ and s_h is the first nontrivial flight in both H and $\hat{H}$. Similarly, let $t = cei(H)$. Then $t = cei(H) = cei(S_k) = cei(\hat{H}) = cei(\hat{S}_r)$ and s_t is the last nontrivial flight in both H and $\hat{H}$. Thus, $1 \leq h \leq t \leq q$. If $x \in \bar{Q}_H$, then $du(H) = x(s_q) + et(s_q) - x(s_1)$ and $dt(H) = x(s_t) + et(s_t) - x(s_h)$. If $y \in \bar{Q}_{\hat{H}}$, then $du(\hat{H}) = y(s_q) + et(s_q) - y(s_1)$ and $dt(\hat{H}) = y(s_t) + et(s_t) - t(s_h)$.

Let $a = cei(S_1) = cei(S(n_1, a_1))$ and $b = csi(S_k) = csi(S(a_{m-1}, n_m))$. And let $c = cei(\hat{S}_1) = cei(S(n_1, \hat{a}_1))$ and $d = csi(\hat{S}_r) = csi(S(\hat{a}_{m-1}, n_m))$. It follows that

$$1 \leq h \leq a < b \leq t \leq q \text{ and } 1 \leq h \leq c < d \leq t \leq q.$$

Moreover, if $x \in \bar{Q}_H$, then $dt(S_1) = x(s_a) + et(s_a) - x(s_h)$ and $dt(S_k) = x(s_t) + et(s_t) - x(s_b)$. And if $y \in \bar{Q}_{\hat{H}}$, then $dt(\hat{S}_1) = y(s_c) + et(s_c) - y(s_h)$ and $dt(\hat{S}_r) = y(s_t) + et(s_t) - y(s_d)$. Since $\hat{S}_1 \subseteq S_1$ and $\hat{S}_r \subseteq S_k$, we have that

$$c \leq a \text{ and } d \geq b.$$

Let $y = (y_{H^1}, y(a_0), y_{\hat{H}}, y(a_m), y_{H^2}) = (x_{H^1}, x(a_0), y_{\hat{H}}, x(a_m), x_{H^2})$ and we want to show that y is feasible for $\hat{U}$. Note that x is feasible for $\hat{U}$. If $y_{\hat{H}}$ is feasible for $\hat{H}$, and if duty $S(a_0, n_1, \hat{a}_1)$ and duty $S(\hat{a}_{m-1}, n_m, a_m)$, i.e. the first and the last duty of $\hat{H}$, are feasible when timed with y, then y is feasible.

Consider the flight time of $S(a_0, n_1, \hat{a}_1)$ and $S(\hat{a}_{m-1}, n_m, a_m)$. Condition (c) implies that $ft(S(a_0, n_1, \hat{a}_1)) \leq ft(S(a_0, n_1, a_1)) \leq maxFlight$ and $ft(S(\hat{a}_{m-1}, n_m, a_m)) \leq ft(S(a_{m-1}, n_m, a_m)) \leq maxFlight$. Therefore, the maximum flight time constraints on both $S(a_0, n_1, \hat{a}_1)$ and $S(\hat{a}_{m-1}, n_m, a_m)$ are always satisfied in any time-feasible timing of $\hat{U}$.

We will show that, in order for y to be feasible for $\hat{U}$, thus $\hat{H}$ dominates H, it suffices to show the followings:

$$y_{\hat{H}} \text{ is feasible for } \hat{H}; \tag{19}$$

$$y(s_1) \geq x(s_1); \tag{20}$$

$$y(s_h) \geq x(s_h) \text{ and } y(s_c) + et(s_c) \leq x(s_a) + et(s_a); \tag{21}$$

$$y(s_q) \leq x(s_q); \tag{22}$$

$$y(s_t) \leq x(s_t) \text{ and } y(s_d) \geq x(s_b); \tag{23}$$

Since x is time-feasible, $y = (x_{H^1}, x_{a_0}, y_{\hat{H}}, x_{a_m}, x_{H^2})$ is time feasible if (19), (20) and (22) hold. Let us consider the feasibility of duty $S(a_0, n_1, \hat{a}_1)$ and duty $S(\hat{a}_{m-1}, n_m, a_m)$, since the feasibility of other duties in $\hat{U}$ is ensured by the feasibility of x and $y_{\hat{H}}$. For $S(a_0, n_1, \hat{a}_1)$, there are two cases as the following:

If a_0 is a repo-late arc, then $S(a_0, n_1, a_1) = a_0 \cup S_1$ and $S(a_0, n_1, \hat{a}_1) = a_0 \cup \hat{S}_1$. Therefore, the duty start time of $S(a_0, n_1, \hat{a}_1)$ in y is equal to duty start time of

$S(a_0, n_1, a_1)$ in x, which is equal to $x(a_0)$. Note that $y(s_c) + et(s_c) \leq x(s_a) + et(s_a)$ in (21). Therefore, total duty time of $S(a_0, n_1, \hat{a}_1)$ in y is equal to $y(s_c) + et(s_c) - x(a_0) \leq x(s_a) + et(s_a) - x(a_0) \leq maxDuty$, in which the last inequality is because that $S(a_0, n_1, a_1)$ is feasible in x.

If a_0 is not a repo-late arc, then $S(a_0, n_1, a_1) = S(n_1, a_1) = S_1$ and $S(a_0, n_1, \hat{a}_1) = S(n_1, \hat{a}_1) = \hat{S}_1$. Thus (21) implies that total duty time of $S(a_0, n_1, \hat{a}_1)$ in y is equal to $y(s_c) + et(s_c) - y(s_h) \leq x(s_a) + et(s_a) - x(s_h) \leq maxDuty$. Moreover, when a_0 is not a repo-late arc, crew may rest before s_h, the first non-trivial flight of H and $\hat{H}$. We need to ensure that this rest is still feasible in y. Since x is feasible, this rest is feasible if crew start s_h in y later than or equal to the time when crew start s_h in x. This is implied by the inequality $y(s_h) \geq x(s_h)$ in (21).

Therefore, if (21) holds, then maximum duty time and flight time constraints on duty $S(a_0, n_1, \hat{a}_1)$ in timing y are satisfied. Similarly, if (23) holds, we can show that $S(\hat{a}_{m-1}, n_m, a_m)$ is also feasible. Thus (19) through (23) implies that $\hat{H}$ dominates H. In the following, we will find a feasible timing of $\hat{H}$ such that (19) through (23) hold.

It follows from the proof of Theorem 3.1.4 that there exists $x^1 \in \bar{Q}_H$ such that $x^1(s_h) \geq x(s_h)$ and $x^1(s_t) \leq x(s_t)$. Since x^1 also minimizes $du(H)$, we have that $x^1(s_1) \geq x(s_1)$ and $x^1(s_q) \leq x(s_q)$.

Since $x^1 \in \bar{Q}_H$, we know that $x^1(s_h) \leq lcs(H)$. Therefore, $lcs(H) \leq lcs(\hat{H})$ in condition (b) implies that

$$x(s_h) \leq x^1(s_h) \leq lcs(H) \leq lcs(\hat{H}).$$

Similarly, since $x^1(s_t) \geq ece(H) - et(s_t)$, we have that

$$x(s_t) \geq x^1(s_t) \geq ece(H) - et(s_t) \geq ece(\hat{H}) - et(s_t).$$

Comparing $x(s_h)$ to $ecs(\hat{H})$, there are two cases as the following:

- If $x(s_h) \geq ecs(\hat{H})$, then $lcs(\hat{H}) \geq x(s_h) \geq ecs(\hat{H})$. Therefore, there exists $y^1 \in \bar{Q}_{\hat{H}}$ such that $y^1(s_h) = x(s_h)$. Since $dt(\hat{H}) \leq dt(H)$ in condition (a) and y^1 minimizes the duration and duty time of $\hat{H}$, we have that $y^1(s_1) \geq x(s_1)$ and $y^1(s_t) \leq x(s_t)$ which also implies that $y^1(s_q) \leq x(s_q)$. Therefore, (19), (20) and (22) hold for y^1. Moreover, y^1 also minimizes the duty time of $\hat{S}_1$, so we have the following:

$$\begin{aligned} dt(\hat{S}_1) &= y^1(s_c) + et(s_c) - y^1(s_h) \\ &\leq x(s_c) + et(s_c) - x(s_h) \quad (\text{ = duty time of } \hat{S}_1 \text{ in } x) \\ &\leq x(s_a) + et(s_a) - x(s_h) \quad (\text{ since } a \geq c) \\ &= x(s_a) + et(s_a) - y^1(s_h), \end{aligned}$$

  which implies that (21) holds for y^1. Let us set $y^2 = y^1$.

- If $x(s_h) < ecs(\hat{H})$, then there exists $y^1 \in \bar{Q}_{\hat{H}}$ such that $y^1(s_h) = ecs(\hat{H})$. Note that $x(s_t) \geq ece(\hat{H}) - et(s_t)$. Therefore, $x(s_t) + et(s_t) \geq ece(\hat{H}) = y^1(s_t) + et(s_t)$. Thus $y^1(s_t) \leq x(s_t)$, which also implies that $y^1(s_q) \leq x^1(s_q)$. Moreover, since $y^1(s_h) = ecs(\hat{H}) > x(s_h)$ and y^1 minimizes the duration of $\hat{H}$, we have that $y^1(s_1) \geq x(s_1)$. Thus, (19), (20) and (22) hold for y^1.

  If $y^1(s_c) + et(s_c) \leq x(s_a) + et(s_a)$, then (21) holds for y^1. Otherwise, $y^1(s_c) + et(s_c) > x(s_a) + et(s_a) \geq x(s_c) + et(s_c)$, since $a \geq c$. Thus $y^1(s_c) \geq x(s_c)$ and we can define y^2 to be the following:

$$y^2(s_i) = \begin{cases} x(s_i), & \text{if } i \leq c \ ; \\ y^1(s_i), & \text{otherwise.} \end{cases}$$

  We can check that y^2 is feasible for $\hat{H}$, $y^2(s_1) = x(s_1)$, $y^2(s_c) + et(s_c) = x(s_c) + et(s_c) \leq x(s_a) + et(s_a)$, $y^2(s_t) = y^1(s_t) \leq x(s_t)$ and $y^2(s_q) = y^1(s_q) \leq x(s_q)$. Thus, (19) through (22) holds for y^2.

In order to show that (23) holds, let us consider the last duty in $\hat{H}$. Note that $y^2(s_t) \leq x(s_t)$. If $y^2(s_d) \geq x(s_b)$, then (23) hold, and y^2 is feasible. Otherwise, since $y^2(s_d) < x(s_b)$ and $d \geq b$, we can define y^3 to be the following:

$$y^3(s_i) = \begin{cases} x(s_i), & \text{if } q \geq i \geq d \ ; \\ y^2(s_i), & \text{otherwise.} \end{cases}$$

It follows that (19) through (23) holds for y^3. Thus we proved that $\hat{H}$ dominates H.

In the case when either $\hat{S}_1$ or $\hat{S}_r$ is a no-crew duty, we can check that the above arguments still apply. □

The following two corollaries relax the requirement in Theorem 3.2.1 that H and $\hat{H}$ must consist of the same set of duty nodes. Sufficient conditions for $\hat{H}$ to dominate H are given in each case, but detailed proofs are omitted, since they also follow the proof of Theorem 3.2.1.

Let us consider the case when H and $\hat{H}$ have different sets of duties, but have the same set of flights, and flights are in the same order, i.e. $H = (n_1, a_1, n_2, \ldots, n_m) = (S_1, \ldots, S_k) = (s_1, \ldots, s_q)$ and $\hat{H} = (\hat{n}_1, \hat{a}_1, \hat{n}_2, \ldots, \hat{n}_p) = (\hat{S}_1, \ldots, \hat{S}_r) = (s_1, \ldots, s_q)$. Note that condition (c) still implies that $\hat{S}_1 \subseteq S_1$, $\hat{S}_r \subseteq S_k$, $a \geq c$ and $b \leq d$. Moreover, the proof of Theorem 3.2.1 still holds in this case, and we have the same sufficient conditions.

Corollary 3.2.1. *Let* $H = (n_1, a_1, n_2, \ldots, n_m) = (S_1, \ldots, S_k) = (s_1, \ldots, s_q)$ *and* $\hat{H} = (\hat{n}_1, \hat{a}_1, \hat{n}_2, \ldots, \hat{n}_p) = (\hat{S}_1, \ldots, \hat{S}_r) = (s_1, \ldots, s_q)$ *be two distinct feasible paths in* $\mathbb{N}_f$ *that have the same set of flights in the same order. Then* $\hat{H}$ *dominates* H *if the condition (a), (b) and (c) in Theorem 3.2.1 hold.*

Let us consider another case when H and $\hat{H}$ have the same set of flights, but flights are not in the same order, i.e. $H = (n_1, a_1, n_2, \ldots, n_m) = (S_1, \ldots, S_k) = (s_1, \ldots, s_q)$ and $\hat{H} = (\hat{n}_1, \hat{a}_1, \hat{n}_2, \ldots, \hat{n}_p) = (\hat{S}_1, \ldots, \hat{S}_r) = (\hat{s}_1, \ldots, \hat{s}_q)$. Note that $(\hat{s}_1, \ldots, \hat{s}_q)$ is a

permutation of $(s_1, \ldots, s_q)$, so we will use s_i instead of $\hat{s}_i$ to refer to a flight in H or $\hat{H}$.

In the following, we will compare the first duties of H and $\hat{H}$, and show that $\hat{S}_1 \in \hat{H}$ and $S_1 \in H$ have the same set of nontrivial flights.

Suppose that $s_j \in \hat{S}_1$ is a nontrivial flight that is not in S_1. Then flight s_i is contained in duty $S_m \in H$, for some $m > 1$, and crew rest after duty S_1 and before S_m. Let s_i be a nontrivial flight in S_1, and we have that

$$ld(s_j) \geq ed(s_i) + minRest.$$

Moreover, suppose that $s_i \notin \hat{S}_1$. Then

$$ld(s_i) \geq ed(s_j) + minRest.$$

It follows that

$$\begin{aligned} ld(s_i) &\geq ed(s_j) + minRest \\ &\geq ld(s_j) - maxTW + minRest \\ &\geq ed(s_i) + 2 \times minRest - maxTW. \end{aligned}$$

Thus, $maxTW \geq ld(s_i) - ed(s_i) \geq 2 \times minRest - maxTw$, which implies that $maxTW \geq minRest$, contradicting to our assumption about the maximum length of time windows. Therefore, $s_i \in \hat{S}_1$. In other words, we showed that each nontrivial flight in S_1 must be contained in $\hat{S}_1$. Moreover, since $s_j \in \hat{S}_1$ and $s_j \notin S_1$, we have that $ft(S_1) \leq ft(\hat{S}_1) - ft(s_j) < ft(\hat{S}_1)$, contradicting to condition (c) in Theorem 3.2.1. Therefore, $s_j \in S_1$, i.e., each nontrivial flight in $\hat{S}_1$ is also in S_1.

Similarly, $\hat{S}_r$ and S_k also have the same set of nontrivial flights. Note that it does not guarantee that $dt(\hat{S}_1) \leq dt(S_1)$ and $dt(\hat{S}_r) \leq dt(S_k)$, because there may be trivial flights in a duty. However, it implies that y_2 and y_3 in the proof of Theorem 3.2.1 are well defined if needed.

Let $A_H = (s_1, \ldots, s_{csi(H)-1})$, $B_H = (s_{cei(H)+1}, \ldots, s_q)$, $A_{\hat{H}} = (\hat{s}_1, \ldots, \hat{s}_{csi(H)-1})$ and $B_{\hat{H}} = (\hat{s}_{cei(H)+1}, \ldots, \hat{s}_q)$ to take in trivial flights. Following the proof of Theorem 3.2.1, we have the following corollary:

Corollary 3.2.2. *Let $H = (n_1, a_1, n_2, \ldots, n_m) = (S_1, \ldots, S_k) = (s_1, \ldots, s_q)$ and $\hat{H} = (\hat{n}_1, \hat{a}_1, \hat{n}_2, \ldots, \hat{n}_p) = (\hat{S}_1, \ldots, \hat{S}_r) = (\hat{s}_1, \ldots, \hat{s}_q)$ be two distinct feasible paths in $\mathbb{N}_f$ that have the same set of flights. Then $\hat{H}$ dominates H if the condition (a), (b) and (c) in Theorem 3.2.1 and the following condition hold.*

(d) *$dt(\hat{S}_1) \leq dt(S_1)$ and $dt(\hat{S}_r) \leq dt(S_k)$;*

(e) *$du(A_{\hat{H}}) \leq du(A_H)$ and $du(B_{\hat{H}}) \leq du(B_H)$;*

3.2.2.3 Arc Aggregation

Since an arc and its end nodes also form a path in N_f, we can define that an arc dominates another arc when their corresponding paths dominate, and apply the results about path dominance in the previous section to arcs in N_f.

Given a feasible arc, the following theorem considers its end nodes and its corresponding repositioning flight, and determines whether this arc is dominated, thus whether it should be included in N_f.

Theorem 3.2.2. *Let $n_1, n_2 \in N_{dy}$ and $n_1 \neq n_2$. Let q be the repositioning flight from n_1 to n_2, and let us define the following condition:*

$$csi(n_1) \geq 1,\ csi(n_2) \geq 1,\ \text{and a repositioning flight can depart after } \max(lce(n_1) + minRest, le(n_1)) \text{ and arrive before } \min(ecs(n_2) - minRest, ed(n_2)) \tag{24}$$

Then all feasible arcs from n_1 to n_2 must be included in N_f, except for the following cases:

(a): *If q is not trivial and $csi(n_2) < 1$, then only the feasible repo-only arc is included;*

(b): *If q is not trivial, $csi(n_2) = 1$ and (24) holds, then only a repo-only arc is included;*

(c): *If q is trivial, $csi(n_2) \neq 1$, then no arc from n_1 to n_2 is included;*

(d): *If q is not trivial, $csi(n_2) > 1$ and and (24) does not hold, then only the feasible repo-late arc is included;*

(e): *If q is not trivial, $csi(n_2) > 1$ and (24) holds, then no arc from n_1 to n_2 is included;*

Proof. First note that, given n_1 and n_2, at most one case can happen. Moreover, we need to show that in each case, if a feasible arc a is not generated, then it is redundant, i.e. (n_1, a, n_2) is dominated by another feasible path in N_f.

Consider case (a). Let a be a feasible arc from n_1 to n_2 in N_f. We need to show that if a is not a repo-only arc, then (n_1, a, n_2) is dominated by a path in N_f. There are two cases for arc a as the following:

- a is a feasible repo-early arc. Then $n_1 \cup a \cup n_2$ is a feasible duty, and we can check that it is equivalent to $H = (n_1, a, n_2)$. Since a feasible duty is always present in N_f, we have that arc a is redundant.

- a is a feasible repo-late arc. Then it follows from the definition of repo-only arc that a repo-only arc b from n_1 to n_2 is also feasible and is equivalent to a. Since b is generated, we have that (n_1, a, n_2) is dominated by $(n_1, b, n_2) \in N_f$.

Consider case (b). Note that the repo-only arc from n_1 to n_2 is feasible by the condition (24). Let a be the feasible repo-only arc, and let $\hat{H} = (n_1, a, n_2)$. Let b be a feasible repo-early or repo-late arc from n_1 to n_2, and let $H = (n_1, b, n_2)$. We will show that $\hat{H}$ dominates H.

Let $x \in \bar{Q}_{n_1}$ such that $x_{csi(n_1)} = lcs(n_1)$, and let $y \in \bar{Q}_{n_2}$ such that $y_{csi(n_2)} = ecs(n_2)$. Let

$$x_a \in [\max(lce(n_1) + minRest, le(n_1)), \min(ecs(n_2) - minRest, ed(n_2)) - et(a)]$$

, and it follows that (x, x_a, y) is the unique optimal timing of $\hat{H}$ that minimize duration and duty time of $\hat{H}$. Therefore, $dt(\hat{H}) = ece(n_2) - lcs(n_1)$, $lcs(\hat{H}) = ecs(\hat{H}) = lcs(n_1)$ and $ece(\hat{H}) = lce(\hat{H}) = ece(n_2)$.

Since a is a repo-only arc, $ft(S(n_1, a)) = ft(n_1) \leq ft(S(n_1, b))$ and $ft(S(a, n_2)) = ft(n_2) \leq ft(S(b, n_2))$, which implies that condition (c) in Theorem 3.2.1 holds. Moreover, note that $lcs(H) \leq lcs(n_1) = lcs(\hat{H})$ and $ece(H) \geq ece(n_2) = ece(\hat{H})$, which also imply that $dt(H) \geq ece(H) - lcs(H) \geq ece(\hat{H}) - lcs(\hat{H}) = dt(\hat{H})$. Therefore, condition (b) and (c) in Theorem 3.2.1 are satisfied. It follows from Theorem 3.2.1 that $\hat{H}$ dominates H, which proves case (b).

Let $n_2 = (s_1, \ldots, s_m)$. If $csi(n_2) > 1$, then let $S = (s_1, \ldots, s_{csi(n_2)-1})$. Let s_j be the first demand after $s_{csi(n_2)-1}$ in n_2. Thus, $s_j = s_{csi(n_2)}$ or $s_{csi(n_2)+1}$. Let $Q = (s_j, \ldots, s_m)$. Then $n_2 = S \cup c \cup Q$, in which c is a trivial flight if $s_j = s_{csi(n_2)}$. Note that S is a no-crew duty. If $cei(n_2) > csi(n_2)$, then $csi(Q) = 1$. Otherwise, Q is a no-crew duty.

Consider case (c). Let a be the feasible trivial arc from n_1 to n_2, and let $H = (n_1, a, n_2)$. If $csi(n_2) = -1$, then $n_1 \cup n_2$ is a feasible duty in N_f, and we can check that it is equivalent to H. If $csi(n_2) > 1$, then let $\hat{H} = (n_1 \cup a \cup S, c, Q)$ that contains two duties $n_1 \cup a \cup S$ and Q. In $\hat{H}$, c is a repo-only arc if $csi(Q) = -1$, and c is a repo-late arc if $csi(Q) = 1$. We can check that a feasible timing of H is also feasible for $\hat{H}$, and vice versa. Therefore, it follows from Corollary 3.2.1 that $\hat{H}$ is equivalent to H. There are three cases as the following:

- $csi(Q) = -1$. Then c is a repo-only arc, and it follows from case (a) of duty $n_1 \cup a \cup S$ and duty Q that $c \in N_f$. Thus, H is dominated by $\hat{H} \in N_f$;

- $csi(Q) = 1$ and c is a trivial arc. Then arc c from $n_1 \cup a \cup S$ to Q is generated in N_f. Thus, H is dominated by $\hat{H} \in N_f$;

- $csi(Q) = 1$ and c is a repo-late arc. If condition (24) for duty $n_1 \cup a \cup S$ and Q does not hold, then all feasible arcs from $n_1 \cup a \cup S$ to Q are generated. Thus, $\hat{H}$ is in N_f, and $\hat{H}$ dominates H. If condition (24) does not hold, a repo-only arc b from $n_1 \cup a \cup S$ to Q is feasible and is in N_f, thus H is dominated by $\hat{H}$, which is dominated by $(n_1 \cup a \cup S, b, Q) \in N_f$.

Consider case (d). Let a be a feasible arc from n_1 to n_2. We will show that arc a is redundant if it is not a repo-late arc. There are two cases for arc a as the following:

- a is a feasible repo-early arc. Since a is feasible, $n_1 \cup a$ is a feasible duty. It follows that $n_1 \cup a \cup S$ is a feasible duty. Let $\hat{H} = (n_1 \cup a \cup S, c, Q)$, in which c is a trivial or repo-late arc if $csi(Q) = 1$ and is a repo-only arc if $csi(Q) = -1$. Let $H = (n_1, a, n_2) = (n_1, a, S \cup c \cup Q)$. H and $\hat{H}$ are equivalent. Using the same argument as in the proof of case (c), we can show that H is dominated by either $\hat{H}$ or another path in N_f.

- a is a feasible repo-only arc. Let $H = (n_1, a, n_2) = (n_1, a, S \cup c \cup Q)$, and let $\hat{H} = (n_1, a^1, S, c, Q)$ containing three duties n_1, S and Q. Note that although arc a and a^1 correspond to the same repositioning flight, they have different end nodes. We can check that H and $\hat{H}$ are equivalent. Moreover, since $csi(S) = -1$ and a^1 is a repo-only arc, it follows from the case (a) of duty n_1 and S that $a^1 \in N_f$. Furthermore, note that $csi(Q) = 1$ and $csi(S) = -1$. If c is trivial, then it follows from case (d) for duty S and Q that $c \in N_f$. Otherwise, none of the cases (a) through (e) for duty S and Q can be applied, meaning that all arcs from S to Q are in N_f. Therefore, $\hat{H} = (n_1, a^1, S, c, Q) \in N_f$ and $\hat{H}$ dominates H.

Consider case (e). Note that the proof of case (b) holds even if $csi(n_2) \geq 1$. Thus if $csi(n_2) > 1$ and condition (24) holds, then the repo-only arc from n_1 to n_2 is feasible and it dominates other feasible arcs from n_1 to n_2. Moreover, the proof of case (c) implies that this repo-only arc is dominated by another feasible path in N_f. Therefore, all feasible arcs from n_1 to n_2 are redundant in this case. □

3.2.3 Arcs From Duty Node To Demand Node

In this section, similar to arcs between duty nodes in N_f, we will define feasibility of arcs from a duty node to a demand node, and show how to aggregate them.

Let $n_1 \in N_{dy}$, $d_1 \in N_{da}$, $d_2 \in N_{db}$, and let d_0 be the last demand of duty n_1. If d_0 is the last flight of a tour containing n_1, then the airplane of this tour is dropped off after demand d_0.

Definition 3.2.5. *An arc from $n_1 \in N_{dy}$ to $d_1 \in N_{da}$ is feasible if d_1 is the last demand of duty n_1.*

Let us consider the arc from n_1 to $d_2 \in N_{db}$. Note that the repositioning flight from n_1 to d_2 must be nontrivial. Otherwise, it is implied by dropping off the airplane after d_0, i.e. arc $n_1 d_0$. Moreover, demand d_2 can not be assigned to the same crew as duty n_1. Thus, repo-late and repo-only arcs are equivalent in this case, and we will only consider repo-only arcs.

Definition 3.2.6. *Let $n_1 \in N_{dy}$ and $d_2 \in N_{db}$. Let d_0 be the last demand of n_1, and let s be the repositioning flight from d_0 to d_2.*

1. *A* repo-early *arc from n_1 to d_2 is feasible if s is nontrivial, $n_1 \cup s$ forms a feasible duty and $ee(n_1 \cup l) \leq ld(n_2)$.*

2. *Let*
$$t = \begin{cases} \max(ece(n_1) + minRest, ee(n_1)), & \text{if } n_1 \text{ is a crew-duty} \\ ee(n_1), & \text{otherwise.} \end{cases}$$

A repo-only *arc from n_1 to d_2 is feasible if s is non-trivial, and s can depart after t and arrive before $ld(d_2)$.*

Let $H = (n_1, a_1, n_2, \ldots, n_m) = (S_1, \ldots, S_k)$ in which $n_m \in N_{db}$. Note that n_m denotes the end of a tour (no arcs out of N_{db}). Therefore, when considering the dominance of paths ending at a demand node, we only need to consider the crew end time of S_k and do not need to consider the crew start time of S_k. Following the proof of Theorem 3.2.1, we have the following theorem about the dominance of two paths ending at a demand node in N_{db}.

Theorem 3.2.3. *Let $H = (n_1, a_1, n_2, \ldots, n_m) = (S_1, \ldots, S_k) = (s_1, \ldots, s_q)$ and $\hat{H} = (\hat{n}_1, \hat{a}_1, \hat{n}_2, \ldots, \hat{n}_p) = (\hat{S}_1, \ldots, \hat{S}_r) = (s_1, \ldots, s_q)$ be two distinct feasible paths in $\mathbb{N}_f$ in which $n_m = \hat{n}_p \in N_{db}$. Then $\hat{H}$ dominates H if all the followings hold.*

(a) *$dt(\hat{H}) \leq dt(H)$;*

(b) *$lcs(\hat{H}) \geq lcs(H)$ and $ece(\hat{H}) \leq ece(H)$;*

(c) *$ft(\hat{S}_1) \leq ft(S_1)$;*

For $d_2 \in N_{db}$, $ets(d_2)$ will be defined later in section 3.3.3. At the high level, $ets(d_2)$ means the following: in any feasible schedule, as long as a tour in this schedule drops off the airplane before time $ets(d_2)$, later part of this schedule can still be the same and total cost does not increase. In other words, it is equivalent that a tour drops off its airplane before demand d_2 at time $ets(d_2)$ or at time $ets(d_2) - \delta$.

Theorem 3.2.4. *Let $n_1 \in N_{dy}$ and $d_2 \in N_{db}$. Let q be the repositioning flight from n_1 to d_2. Then all feasible arcs from n_1 to d_2 are included in N_f, except for the following cases:*

(a) *if $d_2 \notin D_c$, then no arcs from n_1 to d_2 is included in N_f;*

(a) *if q is non-trivial and $csi(n_1) = -1$, then only feasible repo-only arc is included;*

(b) *if q is non-trivial, $csi(n_1) \geq 0$, and it is feasible for q to depart after $max\{lce(n_1) + minRest, le(n_1)\}$ and arrive before $ets(d_2)$, then only feasible repo-only arc is included.*

Proof of Theorem 3.2.4 follows from the proof of Theorem 3.2.2.

The case for paths ending at a demand node in N_{da} is trivial, since the arcs from duty nodes to N_{da} are trivial. Given paths ending at a demand node in N_{da}, we can delete their last arc and consider their dominance using results for paths consisting of only duty nodes.

3.2.4 Cycles In N_f

We have seen that there may be cycles in the demand graph G_f. Similarly, it's possible for the duty network $\mathbb{N}_f$ to contain cycles. We'll show that if there's an upper bound on the time window, then $\mathbb{N}_f$ is acyclic, or only contains specific types of cycles.

Let $minMT$ be the minimum duration of a maintenance request or appointment, and let $minFT$ be the minimum duration of a nontrivial flight leg. Let $maxDTW$ be the maximum length of the time window of a customer-requested demand, and let $maxMTW$ be the maximum length of the time window of a maintenance request or appointment. Let $minMTurn$ be the minimum turn time after a maintenance request or appointment.

Theorem 3.2.5. *If $maxMTW \leq maxDTW$, $maxDTW < minRest + minFT$ and $maxMTW \leq minTurn + minFT + minMT$, then the duty network N_f is acyclic.*

Proof. Let us first show that there is no cycle containing two nodes.

Let n_1 and n_2 be two duty nodes in N_f. Note that a crew duty contains at leat one nontrivial flight, therefore if n_i, $i = 1$ or 2, is a crew duty, then

$$dt(n_i) \geq minFT. \tag{25}$$

If n_i, $i = 1$ or 2, is a non-crew duty, then since a non-crew duty contains at least one trivial flight, we have the following:

$$du(n_i) \geq minMT. \tag{26}$$

Moreover, time window of a duty can not be larger than the time window of any flight in the duty. Therefore, for n_i, $i = 1$ or 2, the following holds:

$$ld(n_i) - ed(n_i) \leq \begin{cases} max(maxDTW, maxMTW) = maxDTW, & \text{if } csi(n_i) \geq 1 \text{ ;} \\ maxMTW, & \text{otherwise.} \end{cases} \tag{27}$$

Furthermore, consider the following about the crew time windows of n_i, $i = 1$ or 2,

$$lcs(n_i) - ecs(n_i) \leq maxDTW. \tag{28}$$

Note that the repositioning flight from a demand a to a demand b is feasible as long as it can depart after $ee(a)$ and arrive before $ld(b)$. Therefore, there is no bound on the length of the time window of a repositioning flight. So (28) may not hold in the case when the only nontrivial flight in the duty is a repositioning flight and the time window of the duty is trivial. So we have the following:

$$\text{if } ld(n_i) > ed(n_i), \text{ then } lcs(n_i) - ecs(n_i) \leq maxDTW. \tag{29}$$

Suppose that there is an arc $\overrightarrow{a}$ from n_1 to n_2, and there is an arc $\overleftarrow{a}$ from n_2 to n_1. There are three cases for n_1 and n_2.

Consider the case when both n_1 and n_2 are crew duties. Note that if both n_1 and n_2 have trivial time windows, then either $\overrightarrow{a}$ or $\overleftarrow{a}$ is infeasible, contradiction. Thus, either n_1 or n_2 has nontrivial time windows, and there are two cases as the following:

- Both n_1 and n_2 have nontrivial time windows. Then (29) holds for $i = 1$ and 2,

and we have the following:

$$
\begin{aligned}
& ece(n_1) + minRest \\
\leq\ & lcs(n_2) && \text{(since rest after } n_1\text{)} \\
\leq\ & ecs(n_2) + maxDTW && \text{(from (29))} \\
=\ & ece(n_2) - dt(n_2) + maxDTW \\
\leq\ & ece(n_2) - minFT + maxDTW && \text{(from (25))} \\
\leq\ & lcs(n_1) - minRest - minFT + maxDTW && \text{(since rest after } n_2\text{)} \\
\leq\ & ece(n_1) - dt(n_1) + 2 \cdot maxDTW - minRest \\
& -minFT && \text{(from (29))} \\
\leq\ & ece(n_1) + 2 \cdot maxDTW - 2 \cdot minFT - minRest && \text{(from (25))}
\end{aligned}
$$

It follows from that $maxDTW \geq minRest + minFT$, contradiction. Thus either $\overrightarrow{a}$ or $\overleftarrow{a}$ is not in $\mathbb{N}_f$.

- One of n_1 and n_2 has trivial time window. WLOG, assume that $ed(n_1) = ld(n_1)$. Then (29) holds for n_2, and we have the following:

$$
\begin{aligned}
& ed(n_1) + minRest \\
\leq\ & ece(n_1) + minRest \\
\leq\ & lcs(n_2) && \text{(since rest after } n_1\text{)} \\
\leq\ & ecs(n_2) + maxDTW && \text{(from (29))} \\
\leq\ & ece(n_2) - minFT + maxDTW && \text{(from (25))} \\
\leq\ & ee(n_2) - minFT + maxDTW \\
\leq\ & ld(n_1) - minFT + maxDTW && \text{(since } \overleftarrow{a} \text{ feasible)} \\
=\ & ed(n_1) - minFT + maxDTW
\end{aligned}
$$

It follows from above that $maxDTW \geq minRest + minFT$, contradiction. Thus either $\overrightarrow{a}$ or $\overleftarrow{a}$ is not in $\mathbb{N}_f$.

Now consider the case when one of n_1 and n_2 is a no-crew-duty. WLOG, assume that n_2 is a no-crew-duty. Thus $\overrightarrow{a}$ is a repo-only arc by Theorem (3.2.2), and there

is rest after n_1 and before repositioning to n_2. Let $\mu = 2 \cdot minFt + minTurn + minMT + minRest$, and then by the conditions on $maxDTW$ and $maxMTW$, we have that

$$maxDTW + maxMTW < \mu. \tag{30}$$

It follows that we have the following:

$$\begin{aligned}
& ece(n_1) \\
& \leq ld(n_2) - minTurn - ft(\overrightarrow{a}) - minRest && (\overrightarrow{a} \text{ feasible}) \\
& \leq ld(n_2) - minTurn - minFT - minRest \\
& = le(n_2) - du(n_2) - minTurn - minFT - minRest \\
& \leq ee(n_2) + maxMTW - minMT - minTurn - minFT - minRest && ((27)(26)) \\
& \leq ld(n_1) + maxMTW - (\mu - minFT) && (\overleftarrow{a} \text{ feasible}) \\
& \leq ed(n_1) + maxDTW + maxMTW - (\mu - minFT) && (\text{from } 27) \\
& \leq ecs(n_1) + maxDTW + maxMTW - (\mu - minFT) \\
& \leq (ece(n_1) - minFT) + maxDTW + maxMTW - (\mu - minFT)
\end{aligned}$$

The last inequality holds, because that n_1 is a crew duty, thus containing at least one nontrivial flight. It follows that $maxDTW + maxMTW \geq \mu$, contradicting to (30). Thus either $\overrightarrow{a}$ or $\overleftarrow{a}$ is not in $\mathbb{N}_f$.

The last case is when both n_1 and n_2 are no-crew-duties. It follows from Theorem (3.2.2) that both $\overrightarrow{a}$ and $\overleftarrow{a}$ are repo-only arcs. Let $\nu = minMT + minMTurn + minFT + minTurn$ and we have the following:

$$\begin{aligned}
& ed(n_1) \\
& \leq ld(n_2) - \nu && (\text{since } \overrightarrow{a} \text{ feasible}) \\
& \leq ed(n_2) + maxMTW - \nu && (\text{from } (27)) \\
& \leq ld(n_1) + maxMTW - 2 \cdot \nu && (\text{since } \overleftarrow{a} \text{ feasible}) \\
& \leq ed(n_1) + 2 \cdot maxMTW - 2 \cdot \nu && (\text{from } (27))
\end{aligned}$$

It follows that $maxMTW \geq \nu > minTurn + minFT + minMT$, contradiction. Thus either $\overrightarrow{a}$ or $\overleftarrow{a}$ is not in $\mathbb{N}_f$.

Thus we showed that there is no cycle of two nodes in N_f. Similarly, we can show that there is no cycle of more than two nodes in N_f. □

In practice, $minRest$ is set to be 12 hours and $minTurn$ is set to be 1 hour. Therefore, if $minMT$ and $minFT$ are 30 minutes, it follows from Theorem 3.2.5 that, in order for N_f to be acyclic, the maximum time window for a customer-requested flight is 12.5 hours, and the maximum time window for a maintenance request is 2 hours.

In our instances, the maximum length of a time window for a customer-requested demand is six hours, and maintenance and appointments have very small or no time window. Therefore, the duty network N_f in our instances is acyclic.

If we require that the maximum time window for customer-requested flight and a maintenance request to be the same, then the following corollary follows from the proof of Theorem 3.2.5.

Corollary 3.2.3. *If $maxDTW = maxMTW \leq \lfloor minRest/2 \rfloor$, then each cycle in N_f consists of only no-crew duties.*

Assume that $maxDTW = maxMTW \leq \lfloor minRest/2 \rfloor$, we can show that if a path is time-feasible, then it can not contain both arc $\overrightarrow{a}$ and arc $\overleftarrow{a}$. Suppose not, and WLOG, let us assume that $\overrightarrow{a}$ is before $\overleftarrow{a}$ in the path. By Theorem (3.2.2), both arcs are repo-only arcs, and crews must rest between them. Thus we have the following:

$$ee(n_1) + minRest < ee(n_1) + ft(\overrightarrow{a}) + minRest + ft(\overleftarrow{a}) \leq ld(n_1) \leq le(n_1)$$
$$\leq ee(n_1) + \Delta(n_1) \leq ee(n_1) + minRest/2$$

, contradiction. It follows that any cycle in $\mathbb{N}_f$ is induced by no-crew-duty nodes, and any time-feasible path in N_f is simple.

3.3 Crew Travelling Home

Pilots may need to travel to an airplane and start a tour, and may need to travel back to their home bases after finishing a tour. Crew travelling information is generated by calling an outside flight generation routine, which returns or simulates commercial flights. Crew travelling is important, because it contributes to both total cost and associated pilot's duty time.

In this section, we will discuss the issues related to crew travelling back to home bases. In particular, we will generate a set of feasible candidate flights for crews to travel home, and introduce the time window for crew travelling home. When considering pickup arcs in section 3.4, we will discuss the more complex problem of crew travelling to pick up an airplane.

One key characteristic of crew travelling is that crew travelling is not "continuous", as opposed to the case of repositioning flight. If a repositioning flight can start at time t_1 and t_2, $t_2 > t_1$, then it can start at anytime in the interval $[t_1, t_2]$, and the flight time and cost are assumed to be the same. But this does not hold for commercial flights. Given an origin and destination, commercial flight schedules may depend on the departure date and are subjected to change, and flights may have different durations and prices. This key characteristic of crew travelling requires us to handle crew travelling differently.

3.3.1 Feasible Travel

In this section, we will introduce some notation related to crew-travel and define feasible travel.

Let C_h be the set of pilots whose tour end time is within the current planning horizon, thus possibly going back to their home bases after their tours. Let $c \in C_h$. If we want to assign pilot c to a tour, we need to determine whether c can arrive home on time after this tour and calculate cost of the travel. Note that if pilot c can work

overtime, then c may be scheduled to arrive home after $te(c)$, but an overtime cost must be calculated and added to the total cost of this tour.

More specifically, a pilot $c \in C_h$ specifies an integer $\gamma_c \geq 0$, which means that the overtime for pilot c is at most $\gamma_c \cdot minOT$. In other words, it is considered feasible if pilot c arrives at c's home base at time $t \in [te(c), te(c) + \gamma_c \cdot minOT]$. And the overtime cost is an increasing function on $\lceil \frac{t-te(c)}{minOT} \rceil$. In practice, $minOT$ is set to be 12 hours, γ_c is at most 6, and the overtime cost function is a linear function.

A *travel* r is define by $(oa(r), da(r), st(r), et(r), c(r))$, in which $oa(r)$ is the origin airport of *travel* r, $da(r)$ is the destination airport, $st(r)$ is travel start time, $et(r)$ is travel end time, and $c(r)$ is the travel cost including overtime cost.

Travel r also contains a commercial flight, which may not depart at $st(r)$. This is because that $oa(r)$ is the airport where a pilot finishes the tour, so it may not be a commercial airport, in which case a pilot needs to take ground transportation from $oa(r)$ to a nearby commercial airport, and $st(r)$ is the time when ground transportation starts. Similarly, $et(r)$ may not be the actual arrival time of the flight.

Information about the commercial flight in *travel* r is determined by the outside flight generation routine, but we only need $st(r)$, $et(r)$ and $c(r)$ from the scheduling point of view. We will assume that $\{st(r), et(r), c(r)\}$ is based on a commercial flight and necessary ground transportation for this flight, and they are returned by the flight generation routine. When there is no confusion, we will omit $oa(r)$ and $da(r)$ in a travel r, since only *travels* with the same start airport and end airport are compared with each other.

Definition 3.3.1. *Given an airport* a *and time* t *after which pilot* c *can start travelling home, travel* r *with* $(a, ba(c), st(r), et(r), c(r))$ *is feasible if* $(st(r), et(r), c(r))$ *is returned by the flight generation routine, and if* $st(r) \geq t$ *and* $et(r) \leq te(c) + \gamma_c \cdot minOT$.

3.3.2 Travel Enumeration

In this section, we will introduce some practical assumptions about travel and present an algorithm which finds feasible and necessary travel efficiently.

For each pilot $c \in C_h$ and each demand d, we will enumerate feasible travels from airport $oa(d)$ and $da(d)$ to the base airport $ba(c)$. It is not practical if we want to enumerate all such feasible travels. Therefore, we will assume the followings:

(**Travel Assumption 1**) all feasible travels between two airports during the current planning horizon have the same travel cost (not including overtime cost).

Under this assumption, we will only need to consider feasible travels that have different overtime costs, resulting in much less number of feasible travels to be considered. Moreover, we will show that we can do further aggregations in some cases.

Given two travels r_1 and r_2, if $st(r_1) = st(r_2)$ and $et(r_1) = et(r_2)$, then r_1 and r_2 have the same overtime cost. So under the Travel Assumption 1, $c(r_1) = c(r_2)$. Therefore, r_1 and r_2 are equivalent, in the sense that they are interchangeable in any feasible schedule.

Thus we will assume that the flight generation routine returns a unique travel with any given start time and end time. Moreover, it is possible that $st(r_1) \leq st(r_2)$ but $et(r_1) \geq et(s_2)$, which implies that r_2 dominates r_1, in the sense that r_1 in any feasible schedule can always be substituted with r_1. In summary, we will assume the following:

(**Travel Assumption 2**) Let r_1 and r_2 be returned by the flight generation routine. Then $(st(r_1) - st(r_2))(et(r_1) - et(r_2)) > 0$.

The reason for the enumeration of feasible travels is to save the computation time when we look for a feasible path in N_f. If travels are not enumerated in advance, we will need to check and generate feasible travels every time we look for a feasible

Algorithm 4 Generate Feasible Travel(c, d, x)

Require: pilot $c \in C_h$, demand $d \in N_{da}$, x: boolean variable
Ensure: T, a set of feasible travels to base $ha(c)$

$T \leftarrow \emptyset$
if $x = 1$ **then**
    $t_1 \leftarrow ee(d)$, $t_2 \leftarrow le(d)$, $a \leftarrow da(d)$
else
    $t_1 \leftarrow at(c)$, $t_2 \leftarrow ld(d)$, $a \leftarrow oa(d)$
for $i = 0$ to γ_c **do**
    if $\exists$ a travel r s.t. $oa(r) = a$, $st(r) \geq t_1$, $et(r) \leq te(c) + i \cdot minOT$ **then**
        generate r_i maximizing $\{st(r) : et(r) \leq te(c) + i \cdot minOT, r \text{ feasible}\}$
        $T \leftarrow T \cup r_i$
        $t_1 \leftarrow st(r_i) + \delta$
        if $st(r_i) \geq t_2$ **then**
            return T
return T

path in N_f. So instead of calling the flight generation routine for the same travel repeatedly, we can look it up in the set of enumerated feasible travels, which is much less time-consuming.

Definition 3.3.2. *Let r and s be two travels with the same origin and destination. r dominates s if the followings hold: $c(r) \leq c(s)$; if a feasible schedule U contains s, then after replacing s with r and keeping $U \backslash s$ to be the same, U is still feasible.*

We will use Algorithm 4 to generate $R_a(c, d)$: a set of dominant feasible *travels* for pilot c to travel home after finishing a demand d, and $R_b(c, d)$: a set of dominant feasible travels for pilot c to travel home before demand d, i.e. after a repositioning flight to demand d.

When x is set to be 1, Algorithm 4 returns $R_a(c, d)$. Otherwise, $R_b(c, d)$ is returned. Note that a maintenance request or an appointment does not require pilots, so it follows from Theorem 3.2.4 that we only need to generate $R_b(c, d)$ for each customer-requested demand d.

In the following, we will show that $R_a(c, d)$ and $R_b(c, d)$ are dominant. Therefore, we only need to consider feasible travels in $R_a(c, d)$ and $R_b(c, d)$ in the scheduling.

Theorem 3.3.1. *If there exists a feasible travel for c to travel home after (or before) demand d, then there is a travel in $R_a(c,d)$ (or $R_b(c,d)$) that dominates it.*

Proof. Let us first consider the case when pilot c travels home after demand d. Since we require d to be a customer-requested flight, the earliest possible time for pilot c to start travelling is $ee(d)$, and pilot c must be able to travel at time $le(d)$.

Note that $R_a(c,d) = T$, in which T is as in Algorithm 4 $(c,d,1)$. Line (6) through Line (12) enumerates feasible travels with different overtime costs, and stops once a feasible travel that can start after $le(d)$ is found. Moreover, Line (10) ensures that the same travel will not be returned at the next iteration, by searching for travels that starts later than the current one. Thus $st(r_i)$ is strictly increasing as i increases.

Let r_k be the last travel added to T. Then $0 \leq k \leq \gamma_c$. Let r be any feasible travel for c to travel home after demand d, and let $m = argmin_{0\leq i\leq\gamma_c}(et(r) \leq te(c) + i \cdot minOT)$. We need to show that there is a travel in $R_a(c,d)$ that is also feasible and has better cost than r. There are two cases as in the following:

- $st(r) \geq le(d)$. We will show that r_k dominates r. Suppose that $st(r_k) < le(d)$. Then $st(r_k) < le(d) \leq st(r)$, so after iteration $i = k$, Line (7) holds because of r. Therefore, Algorithm 4 $(c,d,1)$ will continue after $i = k$, contradicting to the definition of k. Thus $st(r_k) \geq le(d)$, which together with Line (11) imply the following:

  $$k = argmin_{0\leq i\leq\gamma_c}(st(r_i) \geq le(d), et(r_i) \leq te(c) + i \cdot minOT, \text{and } r_i \text{ feasible})$$

  Consider r_m, and we have that $st(r_m) \geq st(r) \geq le(d)$. Therefore, it follows from the above equality of k that $m \geq k$, which implies that $c(r_k) \leq c(r_m) = c(r)$. Moreover, $st(r_k) \geq le(d)$ implies that r_k is always feasible. Thus we have that r_k dominates r.

- $st(r) < le(d)$. We will show that either r_m or r_k dominates r in this case. Suppose that $st(r_k) < st(r)$. Then $st(r_k) < st(r) < le(d)$, so Algorithm 4

$(c, d, 1)$ will not stop at $i = k$, contradiction. So we have $st(r_k) \geq st(r)$. Moreover, if $k \leq m$, then $c(r_k) \leq c(r)$, which together with $st(r_k) \geq st(r)$ imply that r_k dominates r. If $k > m$, then $r_m \in R_a(c, d)$ and $c(r_m) = c(r)$. Moreover, it follows from Line (8) that $st(r_m) \geq st(r)$. Therefore, r_m dominates r.

Note that if $T = \emptyset$, then no travel is feasible, and pilot c can not be assigned a tour ending with d

In the case when pilot c travels home after a repositioning flight leg to d, Algorithm 4 $(c, d, 0)$ is called to generate $R_b(c, d)$. In this case, the earliest possible time for pilot c to start travelling is $at(c)$: available time of pilot c. And c must be able to travel at time $ld(d)$, since the repositioning flight must end before $ld(d)$. Similar to the after-demand case, we can show that $R_b(c, d)$ is dominant. □

3.3.3 Crew Ending Time Of A Tour

Let H be a feasible tour. There are two cases for the last nontrivial flight of H:

(a) a customer-requested demand d;

(b) a repositioning flight to a demand d.

Note that case (a) implies that tour H ends after demand d, i.e. H corresponds to a path in N_f that ends at $d \in N_{da}$. However, case (b) does not mean that H ends before demand d, since it is possible that d is a trivial flight contained in H.

Moreover, after H ends, two things may happen:

1. another tour $\hat{H}$ picks up the airplane of H, so the end time of H will affect the the time when $\hat{H}$ can pick up the airplane;

2. pilot c, who was assigned to H, travels back to home after H, so the crew end time of H will affect the time when c can start travelling.

Let us consider case (a). Note that the end time of H is equal to the crew end time of H in this case. We will show that there exists a time window $[\alpha_d^a, \beta_d^a]$ for crew end time of H such that the followings hold:

- it is not feasible for crew to end H after β_d^a;
- if crew ends H before α_d^a, then $\hat{H}$ can pick up the airplane of H and starts at the earliest possible time, and if needed, each pilot of H can travel back home at the earliest possible time and with the minimum possible cost over all feasible travels for this pilot to travel home after demand d.

Similarly, there exists such a time window $[\alpha_d^b, \beta_d^b]$ in case (b).

In case (a), if we want to assign pilot $c \in C_h$ to H, then we need to determine whether c can travel home after d and the travelling cost. So we need to consider $R_a(c, d)$. If $R_a(c, d)$ is empty, then this assignment is infeasible, since c can not arrive home on time after d.

Otherwise, it follows from Theorem 3.3.1 that if there exists a feasible travel for c to travel home after demand d, then there must exist a feasible travel in $R_a(c, d)$ that dominates it. Note that $R_a(c, d) \neq \emptyset$ does not guarantee that there exists a feasible travel for c. Whether a travel is feasible for pilot c depends on when c can start travelling.

Moreover, in case (a), the airplane assigned to H can only be dropped off at a time between $ee(d)$ and $le(d)$, and the crew assigned to H must end their duty at the same time. In other words, if H ends at time $ee(d) + \delta$, then $\hat{H}$ can pick up the airplane at time $ee(d) + \delta$, and crew can start to travel back home at time $ee(d) + \delta$. Therefore, $[\alpha_d^a, \beta_d^a] = [ee(d), le(d)]$.

Note that if pilot c can start travelling at time $ee(d) + \delta$, then we will choose the feasible travel in $R_a(c, d)$ that has the earliest start time after $ee(d) + \delta$. Therefore, the actual start time of pilot c's travel may not be $ee(d) + \delta$.

Let us consider case (b). In this case, demand d may or may not be contained in H. If $d \in H$, then d must be a trivial flight. Otherwise, d must be a nontrivial flight.

Suppose that $d \notin H$ in case (b). Then H ends with a repositioning flight to d. If $R_b(c,d)$ is empty, then it is infeasible to assign pilot c to H. Otherwise, crew end time of H may be much earlier than $ed(d)$. Therefore, pilot c can start travelling right after the repositioning flight, and do not need to wait until $ed(d)$ or later. This is different from case (a), and it may make an otherwise infeasible assignment to be feasible, or reduce the overtime cost.

Moreover, note that $\hat{H}$, which will pick up the airplane from H, must contain d as its first flight. Therefore, $\hat{H}$ can not start earlier than $ed(d)$. So even if H drops off the airplane much earlier than $ed(d)$, $\hat{H}$ will pick it up and start at time $ed(d)$.

Similarly, suppose that $d \in H$ in case (b). If the crew end time of H is earlier than $ed(d)$, then pilot c can start travelling earlier, and $\hat{H}$ can pick up the airplane from H and start at time $ee(H) = ee(d)$.

In the following, we will define $ets(d)$ for case (b), so that if the crew end time of H is before $ets(d)$, then $\hat{H}$ can pick up the airplane at the earliest possible time: $ed(d)$ or $ee(d)$, depending on whether $d \in H$, and pilot c can travel back to home with the minimum possible overtime cost.

Definition 3.3.3. *Let $R_b(d) = \{r : r \in R_b(c,d), c \in \mathbb{C}_h\}$. The earliest travel start time before d is the following:*

$$ets(d) = \begin{cases} \min\{ed(d), \min\{st(r) : r \in R_b(d)\}\}, & \text{if } R_b(d) \neq \varnothing \\ ed(d), & \text{otherwise.} \end{cases}$$

It follows that $[\alpha_d^b, \beta_d^b] = [ets(d), ld(d)]$.

3.4 Pickup Arcs

In the duty network $\mathbb{N}_f$, A *pickup arc* is an arc from the source node n_s to a duty node or demand node. It contains all information of two pilots picking up an airplane

after possible travelling by commercial flights, and information about the possible repositioning flight leg after travel and before the duty node or demand node.

Comparing to travelling home, when a pilot travels to pick up an airplane, the travelling time is counted toward this pilot's duty time. This difference complicates the underlying optimization problem because of the maximum duty time constraint.

From the scheduling point of view, a feasible tour is represented by total cost, start time of the first and the last flight, and the followings:

(a) demands in this tour;

(b) a pair of pilots assigned to this tour;

(c) an airplane assigned to this tour;

(d) travel to the airplane's available airport, if applicable;

(e) a repositioning flight from the airplane's available airport to the first duty node or demand node, if repositioning is needed;

(f) travel to home bases, if applicable.

Note that we want to construct N_f such that any feasible tour corresponds to a path in $\mathbb{N}_f$ that originates from n_s and ends at a node in $N_{da} \cup N_{db}$. Duty nodes in this path implies (a), the last demand node determines (f), and (b), (c), (d) and (e) are contained in the pickup arc.

In this section, we will generate all feasible and dominant pickup arcs. The reason for the enumeration of pickup arcs is that, in order to determine all information contained in a pickup arc, we need to consider two pilots, an airplane and a duty node all together, and call the outside flight generation routine to generate travels. If feasible pickup arcs are not enumerated upfront, then these computations for each possible pickup need to be done every time we look for a feasible tour in N_f.

Given an airplane, a pair of pilots and their first duty, there may be many choices for pilots to travel, rest, and reposition the airplane to the first duty. Each of these choices may have different cost and timing, and may use different amount of duty time and flight time. In the following part of this section, we will discuss how to generate feasible and dominant pickups, how to optimize them with respect to some chosen objectives, and how to generate the time interval for a feasible pickup in the case when the airplane was available within in a time window.

3.4.1 Available Airplanes

Let a_p be the available airport of the to-be-picked-up airplane, and let $[ed(p), ld(p)]$ be the time interval within which the to-be-picked-up airplane is available to be picked up. There are three possible cases depending on where this airplane comes from:

- *After an airplane.* In this case, an airplane p is picked up when it is available during the planning horizon for the first time, i.e. after time $at(p)$. Therefore, the specific airplane to be picked up is known, $a_p = aa(p)$ and $ed(p) = ld(p) = at(p)$.

- *After a demand* $d \in \mathbb{D}$. In this case, the to-be-picked-up airplane was dropped off after demand d by another tour. Since we do not track individual airplanes for drop-offs and pick-ups (reason for this will be discussed later in the next chapter), the specific to-be-picked-up airplane is not known in this case, unless there is a unique airplane that is type-feasible for demand d. It follows that $a_p = da(d)$, but exact available time of the airplane is not known if d has non-trivial time windows. However, it must be available at time within the interval $[ed(p), ld(p)] = [ee(d), le(d)]$.

- *Before a demand* $d \in \mathbb{D}$. The to-be-picked-up airplane comes from another tour whose last flight is a repositioning flight to demand d. Therefore, $a_p = oa(d)$, but its exact available time is not known, even if d has trivial time windows.

However, since the repositioning flight must end before $ld(d)$, the to-be-picked-up airplane must be available for pickup before $ld(d)$. Moreover, note that if a tour picks up an airplane before demand d, then d must be the first flight of this tour, and pilots who are assigned to this tour must arrive at airport $oa(d)$ before time $ld(d)$. It follows that this tour can not start before $ed(d)$, even if the repositioning flight may drop off the airplane much earlier than $ed(d)$. So we can consider the to-be-picked-up airplane to be available within the interval $[ed(p), ld(p)] = [ed(d), ld(d)]$.

It follows that an airplane is picked up after (or before) a demand only if it has been dropped off after (or demand) the same demand.

Given a pair of pilots and where to pick up the airplane, in order to complete a pickup arc, we need to determine how pilots travel to the airplane and reposition the airplane to their first duty.

We will use the notation *travel-early* to denote the case in which pilot c starts to travel as soon as possible after c is available. Similarly, *travel-late* means that a pilot starts to travel as late as possible to arrive at airport a_p before a given time.

Let a_c be the available airport of pilot c. When considering how pilot c travels to pick up an airplane, we need to consider the following cases: whether c is resting or on-duty (duty time is counting starting from the available time) when available at airport a_c; whether travel early or late; whether put c on rest at airport a_c and a_p.

The repositioning flight from a_p to the first duty or demand node is similar to the repositioning flight between duty nodes. If it is non-trivial, then it may be repo-early or repo-late.

For example, suppose that pilot c was on-duty when available, $a_p \neq a_c$ and there is no need to reposition airplane p to start the first duty node S. We can put c on rest at a_c, then let c travel-early to a_p, and then let c standing by until starting duty S, i.e. no rest after travel and before duty S. Assume that there is another pilot

identical to c. Then in order for this pickup to be feasible for this pair of pilots, the total duty time, which is the actual crew end time of duty S subtracted by the actual travel start time, must be less than $maxDuty$.

One observation about this example is that pilots can travel-late to minimize the total duty time. It also implies that we need to consider the first duty node when generating the travel.

3.4.2 Pickup Graph

In order to generate *pickup arcs*, we will construct a pickup graph PG, whose main purpose is to facilitate the generation process. More specifically, PG consists of three layers of nodes: set of crew nodes N_c, each of which corresponds to a pilot; set of airplane nodes $N_p \cup N_{ad} \cup N_{bd}$, in which each node in N_{ac} corresponds to an available airplane, each node in N_{ad} (N_{bd}) corresponds to a demand after (before) which an airplane may be picked up; set of all duty nodes and demand nodes that are in the duty network N_f.

An arc from a crew node n_c to an airplane node $n_p \in N_p$ is a *crew-arc*, and it means that the pilot c will pick up airplane p when p is available for the first time during the planning horizon. Similarly, there are crew-arcs from crew nodes to nodes in N_{ad} and N_{bd}. An arc from an airplane node to a duty node is an *airplane-arc*, and it means that this airplane will serve demands in this duty. An airplane arc to a demand node means that this airplane will be repositioned to this demand and will be dropped off before this demand.

A crew-arc from a crew node n_c to an airplane node includes pilot c's activities after being available and before the first flight in the planning horizon: whether c rests at a_c; travels early or late; whether c rests at a_a before the first flight.

An airplane-arc corresponds to a repositioning flight which may be trivial. Similar to the arcs between duty nodes in N_f, if the repositioning is not trivial, then there

may be multiple airplane-arcs, depending on whether pilots rest before or after the repositioning flight.

Note that whether a pilot rests at airport a_p before the repositioning flight is included in the crew-arc. Moreover, for a crew of two pilots, it is possible that one pilot rests at a_p, but the other does not. So we only consider two types of airplane-arcs: repo-late and repo-early, but no repo-only.

Recall that we assume that all crew pairs (a crew pair is a pair of pilots) are given as input. Therefore, in order to check and generate a feasible pickup with pilots c_1 and c_2, an airplane node n_p and a duty S, we need to consider the crew arcs from n_{c_1} and n_{c_2} to n_p, and the airplane arc from n_p to duty node S. Note that there may be several feasible pickup arcs with the same pilots, airplane and duty, and they have different timings for the repositioning leg and travel.

An important question is that why we do not combine the pickup graph PG with the duty network $\mathbb{N}_f$, so that a pickup arc corresponds to a path that starts from a pair of crew nodes, through an airplane node, and to the first duty node or a demand node. A feasible tour still corresponds to a path in this new network, and a dynamic programming algorithm, such as a label-setting algorithm, can be used to find feasible tours in this network. Although there will be repeated queries for the crew travel, thus more computation time, we can save memories comparing to the upfront enumeration of pickup arcs.

The reason is that travel in crew arcs and repositioning flight in the airplane arc may depend on each other and may depend on the first duty node or demand node. Therefore, if a label-setting algorithm is used, when considering the labels on the duty node or demand node, we may need to go backward and adjust the crew travel and repositioning flights, in order to make the pickup feasible or minimize total flight time and duty time of the pilots or minimize the overtime cost.

3.4.2.1 Crew-arcs in PG

In this section, we will consider how to generate and aggregate crew-arcs.

Feasibility of a *crew-arc* is determined by the following: feasibility of travelling from airport a_c to a_p with the given time constraints; duty time constraints; if the airplane is picked up before a demand d, then the crew must be able to arrive at airport a_p before time $ld(d)$.

Definition 3.4.1. *Given two crew-arcs a and b that have the same end nodes, a* dominates *b if the followings hold: if b is feasible for a tour, then a is also feasible for the same tour; total cost of a is no more than total cost of b.*

Consider the case in which pilot c is on duty when available at airport a_c, and c travels to pick up an airplane that is available at airport a_p. In this case, let us define the following types of crew-arcs:

(a): No rest at a_c, travel early, and then rest at a_p;

(b): Rest at a_c, travel early, and then rest at a_p;

(c): Rest at a_c, travel late, and then no rest at a_p;

(d): No rest at a_c, travel early, and then no rest at a_p.

Let r_a be the travel in crew-arc (a). Similarly defined are r_b, r_c and r_d.

Note that if there exists a feasible travel, then there must exist a feasible early-travel and a feasible late-travel.

When generating the late-travel in a crew-arc, if the time before which pilots must arrive at airport a_p is not available yet, we will check feasibility of the early-travel and generate a feasible early-travel instead. This early-travel will be updated to a proper late-travel when generating pickup arcs. Therefore, travel r_c in crew-arc (c) is not fixed and depends on the airplane arc.

Algorithm 5 Generate Crew Arcs(n_c, n_p)

Require: crew node n_c for pilot c and airplane node n_p
Ensure: T, a set of feasible crew arcs from n_c to n_p

```
T ← ∅
if travel (a) is feasible then
    T ← T ∪ (a)
    if travel (a) arrives before ed(p) then
        return T
if travel (b) is feasible then
    T ← T ∪ (b)
    if travel (b) arrives before ed(p) then
        return T
if travel (c) is feasible then
    T ← T ∪ (c)
if travel (d) is feasible then
    T ← T ∪ (d)
return T
```

Theorem 3.4.1. *Given a crew node n_c and an airplane node n_p, Algorithm 5 (n_c, n_p) returns T which dominates all other feasible crew arcs from n_c to n_p.*

Proof. Note that if pilot c starts to travel before c's tour start time, then we need to consider possible overtime cost. However, since c is on duty when available, an overtime cost has been calculated when c starts to count duty time earlier. So we do not need to count the overtime cost again.

Let us consider how pilot c rests at airport a_c and a_p. There are four cases in total:

1. No rest at a_c and rest at a_p. Suppose that there is a feasible crew-arc e, in which the travel r_e from a_c to a_p is not an early-travel. To show that (a) dominates e, we only need to show that (a) is also feasible, and (a) ends no later than e.

   Note that pilot c will rest after travel in this case. So in order for crew-arc (a) to be feasible, we need to check the total duty time of pilot c before the rest at a_p. Since pilot c is on duty when available, the time when pilot c starts the duty is already fixed and is the same for both crew-arcs. Let us consider the time

when pilot c ends the duty, i.e. the arrival time of the travel to airport a_p. Since travel r_a starts earlier than r_e, it arrives earlier than r_e following from Travel Assumption 2. Thus the total duty time before rest in crew-arc (a) is less, which implies that crew-arc (a) is also feasible. It follows that (a) dominates e in this case.

2. Rest at both a_c and a_p. Similar to crew-arc (a), since travel r_a starts earlier, it arrives earlier. So we only need to consider crew arc (b) in this case.

3. Rest at a_c and no rest at a_p. It follows that the travel, possible repositioning leg after the travel and the first duty are included in the same duty, and start time of this duty is the travel start time. Suppose that e is a a feasible crew-arc, in which the travel r_e is not a late-travel. Let t be the actual start time of the repositioning flight in e, and let r_c be the late-travel before t. It follows that r_c starts no earlier than r_e, thus crew-arc (c) is also feasible. Therefore, crew-arc (c) dominates other crew-arcs in this case.

4. No rest at both a_c and a_p. In this case, the travel, possible repositioning leg after the travel and the first duty S are also contained in the same duty. But start time of this duty is fixed, since pilot c is on duty when available. In order to show that crew-arc (d) dominate other feasible crew-arcs in this case, it suffices to show that (d) ends no later that any feasible crew-arc, which is implied by the fact that r_d is an early-travel.

Therefore, given how pilot c rests at airport a_c and a_p, crew-arc (a), (b), (c) or (d) is dominant. Let $U = \{(a), (b), (c), (d)\}$. Note that Algorithm 5 (n_c, n_p) returns U, except when Line (4) or Line (8) holds.

If Line (4) holds, then pilot c is available after rest and before $ed(p)$ at airport a_p, and any flight assigned to airplane p can not start earlier than $ed(p)$. Therefore, by

definition, (a) dominates all other feasible crew-arcs from n_c to n_p. Similarly, if Line (8) holds, then crew-arc (b) dominates all other crew-arcs. □

Moreover, in the following, we will show that if Line (9) in Algorithm 5 is executed, then travel-arc (a) is not feasible, which implies that T consists of only travel-arc (b) after the algorithm exits at Line (9).

Suppose that crew-arc (a) is feasible and Line (9) is executed. It follows that Line (4) does not hold. Therefore, we have that

$$et(r_a) + minRest > ed(p). \tag{31}$$

Moreover, note that pilot c rests then travels in crew-arc (b), and does not rest in crew-arc (a). Therefore, $st(r_a) \leq st(r_b)$, which implies that $et(r_a) \leq et(r_b)$. Since Line (8) holds, we have that $et(r_b) + minRest \leq ed(p)$. Thus, $et(r_a) \leq et(r_b) \leq ed(p) - minRest$, contradicting to (31). Thus if Line (9) is executed, then travel-arc (a) is not feasible.

In the case when pilot c is on rest when available at airport a_c, there are two sub-cases depending on whether pilot c rests again at airport a_p. Let us consider the following two types of crew-arcs:

(e): Travel early and rest at a_p;

(f): Travel late and no rest at a_p.

Similar to the proof of Theorem 3.4.1, if c rests after travel, then early-travel dominates. Otherwise, late-travel dominates. However, pilot c may start the first ever duty in this case. So if the travel start time is earlier than $ts(c)$, then we need to consider overtime cost. And if crew-arc (e) has overtime cost, we will minimize the overtime cost when generating pickup arcs.

Corollary 3.4.1. *If crew-arc (e) has no overtime cost and* $et(r_e) + minRest \leq ed(p)$*, then crew-arc (e) dominates all other feasible crew-arcs.*

3.4.2.2 Airplane-arcs in PG

In this section, we will consider how to generate and aggregate airplane-arcs and define the time interval for the start time of an airplane-arc.

There are two types of airplanes arcs from an airplane node n_p to a duty node S: repo-early, in which pilots rest after repositioning flight and before S; repo-late, in which the repositioning leg is combined with duty S to form a new duty.

Note the time window for the available time of the airplane is $[ed(p), ld(p)]$, and the time window of duty S is $[ed(s), ld(S)]$. So we can generate the airplane-arc in the same way as generating arcs between duties. Moreover, similar to Theorem 3.2.2, we have the following:

Theorem 3.4.2. *Feasible repo-early and repo-late airplane-arcs from n_p to S are generated except for the following cases:*

(a): if the repositioning is trivial and $csi(S) \neq 1$, then no airplane-arcs are generated;

(b): if the repositioning is non-trivial and $csi(S) > 1$, then only feasible repo-late airplane-arc is generated;

There is at most one airplane arc from an airplane node n_p to a demand node d: repo-early, in which pilots start the repositioning as soon as possible.

In the remaining part of this sub-section, we will consider airplane-arcs to a duty node in N_f. The case for airplane-arcs to a demand node follows similarly.

Let a_3 be an airplane-arc from n_p to duty S, and let s be the repositioning flight in a_3. Let us define the duration of a_3 as the following:

$$du(a_3) = \begin{cases} du(s) + minRest, & \text{if } a_3 \text{ is repo-early and } S \text{ is crew-duty} \\ du(s), & \text{otherwise.} \end{cases}$$

The actual start time of repositioning flight s has two effects on the later part of

the tour:

$$\text{start time and crew start time of duty } S; \tag{32}$$

$$\text{start time of crew travelling home after } S. \tag{33}$$

We will generate a time interval $[ed(a_3), ld(a_3)]$ for the start time of the repositioning leg s so that: both objectives, (32) and (33), are minimized, if s starts before $ed(a_3)$; it is not feasible for s to start after $ld(a_3)$.

Note that here we only consider end nodes of arc a_3, n_p and S, i.e. the to-be-picked-up airplane and the first duty. So whether the repositioning flight s can actually start at time t, $ed(a_3) \leq t \leq ld(a_3)$, also depends on whether pilots are available at time t. In other words, it depends on crew-arcs. This will be considered later when we combine crew-arcs and airplane-arcs to generate feasible pickup arcs.

Definition 3.4.2. *Let S ba a duty and let d be the first flight of S. Define the early crew cutoff time of S to be the following:*

$$co(S) = \begin{cases} ecs(S), & \text{if } S \text{ is a crew-duty} \\ \min\{ed(S), ee(S) - minRest, ets(d)\}, & \text{otherwise.} \end{cases}$$

We will show that the time interval for the start time of the repositioning flight s defined in the following is well-defined:

$$[ed(a_3), ld(a_3)] = [\max\{ed(p), co(S) - du(a_3)\}, ld(S) - du(a_3)]. \tag{34}$$

Theorem 3.4.3. *Let s be the repositioning flight in the airplane arc a_3. Then it is infeasible to start s after time $ld(a_3)$, and (32) and (33) are minimized if s starts at time $ed(a_3)$.*

Proof. There are two cases depending on whether S is a crew-duty.

Consider the case when S is a crew-duty. Note that after the repositioning flight s, airplane p is immediately available. So if s starts at time t, then airplane p will be

ready for duty S at time $t + du(s)$. However, crew may need to rest after s, in which case they will be ready for duty S later than the ready time of airplane p. Moreover, note that it is possible that $ed(S) = ld(S)$, but $ecs(S) \neq lcs(S)$, so the actual crew start time of S may need to be considered separately.

There are two subcases with respect to a_3 as follows:

- a_3 is a repo-late arc. Since $du(a_3) = du(s)$, both airplane and crew will be ready at time $t + du(s)$. If $ed(S) \leq t + du(s) \leq ld(S)$, then start time of S is $t + du(s)$, and crew start time of S is $ecs(S) + (t + du(s) - ed(S))$, which implies that the crew start time of S does not need to be considered separately, and it is fixed if t is given.

- a_3 is a repo-early arc. In this case, crew needs to rest after s and before duty S. Since S is a crew-duty, $du(a_3) = du(s) + minRest$ from the definition of $du(a_3)$. Therefore, crew will be available at time $t + du(a_3) = t + du(s) + minRest > t + du(s)$. Note that $csi(S) = 1$, following from Theorem 3.4.2. Therefore, duty start time of S and crew start time of S are the same, and if $ed(S) \leq t + du(a_3) \leq ld(S)$, then both start time of S and crew start time of S are $t + du(a_3)$.

Moreover, since S is a crew-duty, if pilots need to travel back home after S, they can start travelling at the crew end time of S, which only depends on the start time of S. It follows that (32) and (33) are minimized, if s starts before or at $ed(S) - du(a_3)$, and

$$[ed(S) - du(a_3), ld(S) - du(a_3)]$$

is the desired time interval.

Furthermore, a repositioning flight can only start after the airplane is available. Therefore, if $ed(p) > ed(S) - du(a_3)$, then this pickup is infeasible. Otherwise, we

have the following:

$$[ed(a_3), ld(a_3)] = [\max\{ed(p), ed(S) - du(a_3)\}, ld(S) - du(a_3)].$$

Now consider the case when S is a no-crew-duty. It follows that a_3 is a repo-early arc, and $du(a_3) = du(s)$. There are two subcases as in the following.

If S is the last duty of the tour, then pilots may need to travel back home after S. Since S does not require pilots, pilots can start travelling immediately after the repositioning flight s. Let d be the first flight of duty S. It follows from the definition of $ets(d)$ that if s starts earlier than $ets(d) - du(a_3)$, then any pilot can travel back home without overtime costs.

If S is not the last duty of the tour, then pilots rest after s. And if pilots can finish resting before $ee(S)$, then after duty S, pilots will be available before $ee(S)$. Therefore, if s starts before $\min\{ed(S), ee(S) - minRest\} - du(a_3)$, then both airplane and pilots will be available at time $ee(S)$, which is the earliest possible.

Thus in the case when S is a no-crew-duty, the desired interval is

$$[\max\{ed(p), \min\{ed(S), ee(S) - minRest, ets(d)\} - du(a_3)\}, ld(S) - du(a_3)],$$

which completes the proof. □

3.4.3 Optimizing Pickup Arcs

In this section, after generating crew-arcs and airplane-arcs, we will consider how to generate all feasible and necessary pickup arcs and how to optimize them.

Let c_1 and c_2 be two pilots who will form a feasible crew, let n_p be an airplane node and let S be a duty node. For $1 \leq i \leq 2$, let a_i be a crew-arc from c_i to n_p and let r_i be the travel in a_i. Let a_3 be the airplane arc from n_p to S. WLOG let's assume that both pilots need to travel to airport a_p to pick up the airplane p, and the repositioning flight from a_p to duty S is nontrivial.

If a pilot travels late or travels early but has overtime cost, then this pilot's travel is not *fixed*, and it depends on the airplane arc a_3. For example, if both crew-arcs a_1 and a_2 are travel-late arcs, and the repositioning flight in a_3 has been fixed, then it is more beneficial to let pilots travel to arrive not only before the start time of the repositioning flight, but also as close to it as possible. Therefore, when checking the feasibility and generating a pickup arc, we may be able to adjust travel and the repositioning flight, so that the pickup is not only feasible, but also optimal.

In general, given $\{a_1, a_2, a_3\}$, we will find timings of $\{r_1, r_2, s\}$, so that the pickups with these timings are not only feasible, but also optimal with respect to the following objectives:

(a): minimize the end time and crew end time of duty S;

(b): minimize the amount of used duty time after duty S, if S is a *crew-duty*;

(c): minimize the overtime cost.

Note that it is possible that a pair of pilots have different accumulated duty times. In this case, the larger one is defined as the duty time of this pair. The flight time of a pair of pilots is defined similarly.

Objective (a) and (b) in the above are necessary, because there may be a repo-early arc after duty S, and pilots may need to travel back home after finishing duty S. Note that, because the total used flight time of each pilot after S is fixed when arcs $\{a_1, a_2, a_3\}$ are given, we don't need to minimize total used flight time.

However, it may be impossible to minimize objective (a), (b) and (c) simultaneously. Objective (a) is related to whether the connection after S is feasible, or whether pilots can travel back home on time if needed. Objective (b) is used to ensure that the duty, induced by S and the arc after S, has the minimum total duty time and is feasible. Objective (c) is related to the total cost. Therefore, in the case when it is

not possible to find a minimum with respect to all three objectives, we will use the priority $(a) \geq (b) \geq (c)$.

Note that feasibility of a pickup can be checked easily. The procedure of optimizing a pickup arc stops, once infeasibility is detected. In the remaining part of this section, we will omit the feasibility-checking steps and focus on the optimizing steps.

Let e be a feasible pickup arc induced by $\{a_1, a_2, a_3\}$, and let $x = \{x(r_1), x(r_2), x(s)\}$ be an optimal timing for e. Note that $ed(p) \leq ed(a_3) \leq x(s) \leq ld(a_3)$ and the to-be-picked-up airplane may be available at anytime in the time interval $[ed(p), ld(p)]$.

We will show that there exist $du(e)$: duration of e, and $[ed(e), ld(e)]$: *time window* of e, which is contained in $[ed(p), ld(p)]$, so that if the to-be-picked-up airplane is available at time t, $ed(e) \leq t \leq ld(e)$, then $x = \{x(r_1), x(r_2), t\}$ is optimal with respect to objective (a), (b) and (c), and crew end time after duty S is $t + du(e)$. Note that $x(r_1)$ and $x(r_2)$ are fixed, and the optimality of x may be approximated in some cases.

A key observation of the time window of pickup arcs is that it is for airplanes, not for pilots. This is because that travel is not continuous. In summary, our procedure will find optimal pickup arcs and their durations and time windows, and in an optimal pickup arc, each pilot's travel is fixed.

3.4.3.1 Sub-cases

Given crew nodes $\{n_{c_1}, n_{c_2}\}$, airplane node n_p and a duty node S, generation procedure of pickup arcs associated with them is divided into six sub-cases based on the airplane-arcs and crew-arcs.

Let a_3 be the airplane-arc from n_p to S, and let $\{a_1, a_2\}$ be the crew-arcs from n_{c_1} and n_{c_2} to n_p respectively.

If a_3 is a repo-late arc, then a_3 is considered to be *fixed*, in the sense that the repositioning flight s only needs to start between $ed(a_3)$ and $ld(a_3)$ in order to minimize

objective (a), (b) and (c). However, this does not mean that it is not feasible for s to start before $ed(a_3)$. Since a_3 is a repo-late arc, it follows that S is a crew-duty and s and S are contained in the same duty. Therefore, it does not help the objective (a), (b) and (c) to let s start earlier that $ed(a_3)$.

If a_3 is a repo-early arc, then pilots rest after repositioning flight s. So for $1 \leq i \leq 2$, s is contained in the duty induced by crew-arc a_i and s. Therefore, s needs to start as soon as possible once pilots c_1, c_2 and airplane p are available at airport a_p, with additional considerations of minimizing the possible overtime cost. In this case, s depends on the crew-arcs, thus not *fixed*.

Consider the crew-arc a_i, $1 \leq i \leq 2$. If a_i is a travel-early arc and there is no overtime cost, then we consider crew-arc a_i as *fixed*. More specifically, since a_i is a travel-early arc, pilot c_i is either on duty when available at airport a_{c_i}, or on rest at a_{c_i} and will rest again at a_p. In both cases, travelling later than r_i will not help the objective (a), (b) and (c), or make an otherwise infeasible pickup to be feasible.

Note that if crew-arc a_i is not *fixed*, then pilot c_i must be on rest at airport a_{c_i}, so c_i can travel later if needed.

Given two crew nodes, an airplane node and a duty node, it follows from above that there are six cases for a *pickup arc* associated with them: how many crew-arcs are fixed, combined with whether the airplane arc is fixed.

In general, our procedure of generating and optimizing pickup arcs considers how to time the fixed arc(s) first, then tries to time the no-fixed arc(s) to minimize objective (a),(b) and (c). When all arcs are not fixed, we will time them using the priority of the objectives.

3.4.3.2 Example

In this section, we will illustrate one sub-case as an example, and other five sub-cases follow similarly. Let us consider the case when both a_1 and a_2 are not *fixed*, but arc

a_3 is *fixed*.

Since a_3 is fixed, the repositioning flight must start after time $ed(a_3)$ and before time $ld(a_3)$. So in order to minimize objective (a), (b) and (c), both travels need to arrive as close as possible to the actual repositioning start time.

Moreover, although both a_1 and a_2 are not fixed, we generated early-travels for them when generating crew-arcs. Therefore, for $1 \leq i \leq 2$, let t_i be the available time of pilot c_i at airport a_p after the early-travel and possible rest implied by the crew-arc.

Note that it is possible that $t_1 \neq t_2$, so let us define $t = \max\{t_1, t_2\}$.

If $t > ld(a_3)$, then this pickup is not feasible. Otherwise, consider the following interval for the repositioning start time:

$$[ed(r), ld(r)] = [\max\{t, ed(a_3)\}, \max\{t, ed(a_3), \min(ld(a_3), ld(p))\}]. \tag{35}$$

In the following, we will show that (35) is well-defined.

First of all, note that no matter how crew travels, it is infeasible for s to start before time t. And considering the airplane arc a_3 and duty S, we have that the earliest start time for s is $ed(a_3)$. Therefore, $ed(r)$ should be the maximum of the earliest airplane available time, earliest crew available time and earliest repositioning start time, i.e.

$$ed(r) = \max\{ed(p), t, ed(a_3)\} = \max\{t, ed(a_3)\},$$

in which the last equality is because of $ed(a_3) \geq ed(p)$. Note that $ed(r) \leq ld(a_3)$.

Now consider $ld(r)$. If $ed(r) \geq \min(ld(a_3), ld(p))\}$, then there are two cases: $ed(r) = ld(a_3)$, thus $ld(r) = ed(r) = ld(a_3)$ since it is not feasible to start s after $ld(a_3)$; $ld(p) \leq ed(r) < ld(a_3)$. In the latter case, since $t \leq ed(r)$, $ld(p) \leq ed(r)$ and $ed(a_3) \leq ed(r) < ld(a_3)$, we know that both crew and airplane can be available before time $ed(r)$ and it is feasible for s to start at $ed(r)$. Moreover, the start time of s

should be no later than $ed(r)$, since objective (a) has the highest priority. Therefore, $ed(r) \geq \min(ld(a_3), ld(p))$ implies that $ld(r) = ed(r)$.

Similarly, if $ed(r) < \min(ld(a_3), ld(p))$, then s needs to start after $ed(r)$ and before $\min(ld(a_3), ld(p))$.

Therefore, (35) is well-defined. Moreover, if $ed(r) < ld(r)$, then

$$ed(p) \leq ed(a_3) \leq ed(r) < ld(r) \leq \min(ld(a_3), ld(p)). \tag{36}$$

Otherwise, $ld(p) \leq ed(r) = ld(r) = \max\{t, ed(a_3)\} \leq ld(a_3)$.

For each pilot c_i, $1 \leq i \leq 2$, we then enumerate all feasible late-travels that arrives within $[ed(r), ld(r)]$, and one feasible travel that arrives just before $ed(r)$. In order to reduce size of the problem or speed up the algorithm, we may only enumerate a subset of travels.

In the set of enumerated travels of both pilots, consider the travel such that its end time is the latest, and let us assume that this travel is associated with pilot c_1. Then for each enumerated travel of c_1, we will take an enumerated travel of c_2 whose arrival time is earlier and is the closest, and generate a pickup arc based on this pair of travels for c_1 and c_2.

Given a pair of travels, let us define

$$[t_c, t_r], \tag{37}$$

and t_c is the available time for this pair of pilots at airport a_p after travel and possible rest, and t_r is the latest start time of repositioning flight s at airport a_p such that duty times of both pilots are feasible. If $t_c > t_r$, then this pickup is not feasible.

Now let us consider $[ed(r), ld(r)]$ and $[t_c, t_r]$. If $t_r < ed(r)$, then this pickup arc is not feasible because of violating total duty time. If $t_c > ld(a_3)$, then this pickup is also not feasible because of the definition of $ld(a_3)$. Otherwise, we will generate a

pickup arc e, and define the followings:

$$\begin{aligned}[ed(e), ld(e)] &= [\min\{t_r, \max(t_c, ed(r))\}, \min\{t_r, \max(t_c, ld(r))\}] \\ &= [\max(t_c, ed(r)), \min\{t_r, \max(t_c, ld(r))\}]; \end{aligned} \tag{38}$$

and

$$du(e) = \begin{cases} du(a_3) + ece(S) - ed(S), & \text{if } S \text{ is a crew-duty} \\ du(s), & \text{otherwise.} \end{cases}$$

It is easy to check that (38) is well-defined. Moreover, if $ed(e) < ld(e)$, then $t_c < ld(r)$ and $[ed(e), ld(e)] = [\max(t_c, ed(r)), \min(t_r, ld(r)]$. It follows that $ed(r) \leq ed(e) < ld(e) \leq ld(r)$. Furthermore, it follows from (36) that if $ed(e) < ld(e)$, then $[ed(e), ld(e)]$ is contained in both $[ed(p), ld(p)]$ and $[ed(a_3), ld(a_3)]$. If $ed(e) = ld(e)$, then $ed(e) \geq ed(r) \geq ed(a_3) \geq ed(p)$.

Therefore, we have the followings: if airplane p is available after time $ld(e)$, then pickup arc e is not feasible; if p is available before $ed(e)$, then the repositioning flight s starts at $ed(e)$; if p is available at $t = ed(e) + \delta$ and $0 \leq \delta \leq ld(e) - ed(e)$, then s starts at $ed(e) + \delta$, and in the following, we will show that the crew end time after duty S is $ed(e) + \delta + du(e)$.

If S is a no-crew duty, then it is trivial, since the crew end time of S is the end time of the repositioning flight leg s. WLOG, let us assume that S is a crew-duty.

If a_3 is a repo-early arc, then $csi(S) = 1$, i.e. $ed(S) = ecs(S)$, and $du(a_3) = du(s) + minRest$. Therefore, if s starts at time $t = ed(e) + \delta$, then s ends at time $t + du(s)$, and duty S can start at time

$$\begin{aligned} & t + du(s) + minRest \\ &= ed(e) + \delta + du(s) + minRest \\ &\geq ed(a_3) + \delta + du(s) + minRest \\ &= co(S) + \delta \\ &= ed(S) + \delta \end{aligned}$$

It follows that the crew end time of S is

$$t + du(s) + minRest + (ece(S) - ed(S)) = t + du(e).$$

If a_3 is a repo-late arc, then $du(a_3) = du(s)$. So if s starts at time $t = ed(e) + \delta$, then s ends at $t + du(s)$. Note that if $ed(S) = ld(S)$, then $t + du(s) = ed(S) = ld(S)$, and crew end time of S is $ece(S)$. Otherwise, $ed(S) \leq t + du(s) \leq ld(S)$. Therefore, in this case, the crew end time of S is $t + du(s) + ece(S) - ed(S) = t + du(e)$.

CHAPTER IV

COLUMN GENERATION

4.1 MIP Formulation and Dual

Let T be the set of all feasible tours. Since each feasible tour corresponds to a path in the duty network N_f, a feasible tour t has a time window $[ed(t), ld(t)]$ for the available time of the airplane assigned to t, and t has duration $du(t)$ such that if the airplane is available at time $ed(t) + \delta$, then tour t picks it up and starts the first flight at time $ed(t) + \delta$, and the end time of tour t is $ed(t) + \delta + du(t)$.

Let V be the set of all vector demands (demands with non-trivial time windows). V_{ad} and V_{bd} are two special sets that have the following meanings: if $p \in V_{ad}$, then p denotes both a vector demand in V and an event that an airplane will be picked up or dropped off after demand p; if $p \in V_{bd}$, then p denotes both a vector demand in V and an event that an airplane will be picked up or dropped off before demand p; Let $U = V_{ad} \cup V_{bd}$.

Similar to V_{ad} and V_{bd}, we define N_{ad}^g and N_{bd}^g, which consist of all demands that are feasible for airplane group g, and let $W^g = N_{ad}^g \cup N_{bd}^g$.

Given the set of all feasible tours, we need to pick a minimum cost subset such that each demand is covered at most once, and if two or more tours use the same airplane, then the connection among them is feasible.

Moreover, let us note the followings: the specific airplane of a tour may not be known, but this airplane's group type is known; there is no penalty for idle pilots or airplanes, but there is penalty for un-covered (therefore chartered) demands; if two or more tours use the same airplane, the later tour must pick up the airplane after it has been dropped off by the earlier tour.

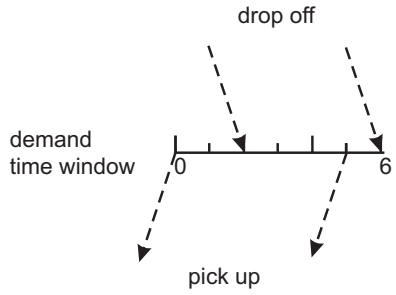


Figure 6: Pick-up time

Furthermore, if the tour that drops off the airplane has non-trivial time windows, then the airplane may be dropped off at any time within the time window. For example, let us consider Figure 6, in which demand d has time window $[0, 6]$, time window for the end time of tour t^1 which contains d as the last flight is $[2, 6]$, and time window for the start time of tour t^2 which picks the airplane from tour t^1 is $[0, 5]$. Therefore, if tour t^1 ends at time 3, then tour t^2 can pick up the airplane and start immediately at time 3. However, if tour t^1 ends at time 5.5, then it is not feasible for tour t^2 to pick up this airplane, since t^2 can not start after time 5.5. In other words, feasibility of this pickup and the actual start time of tour t^2 depend on the actual end time of tour t^1, which is not known in advance and must be incorporated into the formulation.

The decision variables are the followings:

x_t: boolean variable for each feasible tour t denotes whether tour t is in the optimal schedule;

y_d: boolean variable for each demand d denotes whether the demand d is chartered;

$z_{(p,q)}$: non-negative variable for $p \neq q \in U$ denotes the difference of airplane dropped-off time and picked-up time. For example, if $p \in V_{ad}$, $q \in V_{bd}$ and $z_{(p,q)} > 0$, then there exists a tour t^2 in the solution, which picks up an airplane from a tour t^1 after demand p and drops off this airplane before demand q, and tour t^2 has non-trivial time window $[ed(t^2), ld(t^2)]$. Moreover, this airplane must be

dropped off by t^1 at time $ed(t^2)+z_{(p,q)}$, and tour t^2 must pick it up at that time. Note that tour t^2 also must drop off the airplane at time $ee(t^2)+z_{(p,q)}$.

Cost coefficients in the objective function are the followings:

c_t: actual cost of tour t including crew-related cost, and it is assumed that this cost has been calculated for each tour;

c_d: charter cost for each demand d;

The MIP formulation for our problem is the following:

$$Min \quad \sum_{t\in T} c_t x_t + \sum_{d\in \mathbb{D}} c_d y_d \tag{39}$$

$$s.t. \quad \sum_{t\in T} m_{(a,t)} x_t \le 1, \qquad \forall\, a \in \mathbb{A} \tag{40}$$

$$\sum_{t\in T} m_{(c,t)} x_t \le 1, \qquad \forall\, c \in \mathbb{C} \tag{41}$$

$$\sum_{t\in T} m_{(d,t)} x_t + y_d = 1, \qquad \forall\, d \in \mathbb{D} \tag{42}$$

$$\sum_{t\in T} m^g_{(w,t)} x_t \le 0, \qquad \forall\, f \in F, g \in A^f, w \in W^g \tag{43}$$

$$\sum_{t\in T} m_{(u,t)} x_t + \sum_{p,q\in U, p\neq q} m^u_{(p,q)} z_{(p,q)} \le 0, \qquad \forall\, u \in U \tag{44}$$

$$\sum_{t\in T} m_{(p,q,t)} x_t + z_{(p,q)} \le 0, \qquad \forall\, p,q \in U \text{ and } p \neq q \tag{45}$$

$$x_t = \{0,1\}, \forall\, t \in T,\ y_d = \{0,1\}, \forall\, d \in D,$$

$$z_{(p,q)} \ge 0, \forall\, p,q \in U \text{ and } p \neq q.$$

Let us define coefficients in the constraints of the MIP problem (39), and discuss the meanings of constraints (40) through (45).

(40): airplane covering constraints ensure that each airplane is picked up at most once. For an airplane $a \in \mathbb{A}$, $m_{(a,t)} = 1$ if airplane a is picked up by tour t

after being available, and $m_{(a,t)} = 0$ otherwise. Note the difference between an airplane assigned to a tour and an airplane picked up by tour.

(41): crew covering constraints ensure that each pilot is assigned to at most one tour. $m_{(c,t)} = 1$ if pilot c is assigned to tour t, otherwise, $m_{(c,t)} = 0$.

(42): demand covering constraints ensure that each demand is contained in at most one tour. $m_{(d,t)} = 1$ if demand d is contained in tour t, otherwise, $m_{(d,t)} = 0$.

(43): airplane connection constraints ensure that an airplane is picked up only after it was dropped off before the pickup and ensure that this airplane is feasible for the picking-up tour. Note that $W^g = N^g_{ad} \cup N^g_{bd}$.

Suppose that $w \in N^g_{ad}$ for an airplane group $g \in A^f$. Then $m^g_{(w,t)} = -1$ if an airplane of group g is dropped off after demand w by tour t, $m^g_{(w,t)} = 1$ if an airplane of group g is picked up after demand w by tour t, and $m^g_{(w,t)} = 0$ otherwise.

Since demand w can be contained in at most one tour (constraints (42)), there exists at most one tour such that $m^g_{(u,t)} = -1$ in any feasible solution. Suppose that t is such a tour and $x_t = 1$. Note that the right hand side of (43) is equal to 0. Therefore, at most one tour can pick up this airplane.

The case in which $w \in N^g_{bd}$ follows similarly. Note that in this case, it is possible that there exist more than one tours such that $m^g_{(u,t)} = -1$ in a feasible solution. Let t^1 and t^2 be two distinct such tours. Note that the last flight of both t^1 and t^2 is a repositioning flight to demand w. Suppose that both x_{t^1} and x_{t^2} are equal to 1 in an optimal solution, i.e. two airplanes are dropped off before demand w. Then since this airplane can only be picked up by a tour that contains demand w as the first demand, at most one of theses two airplane will be picked up. WLOG, assume that the airplane of tour t^1 is picked up. Then we can delete the

last repositioning flight leg of tour t^2, and the resulting solution is still feasible, but has less cost, contradicting to optimality of the chosen solution. Therefore, at most one tour t such that $x_t = 1$ and $m^g_{(u,t)} = -1$ in an optimal solution in this case.

Note that an airplane can only be in one airplane group. So an airplane can only be dropped off at most once in all constraints (43) for a demand. In the case when an airplane is dropped off, but no other airplanes will pick it up, the summation in the left hand side of (43) is equal to 0 or -1 in an optimal solution, and (43) still holds.

(44): pickup time constraints ensure a feasible pickup when a tour t picks up an airplane that was dropped off after time $ed(t)$. Note that $U = V_{ad} \cup V_{bd}$. Suppose that $u \in V_{ad}$. Then we define $m_{(u,t)} = ee(t) - le(u)$ (≤ 0) if tour t drops off an airplane after demand u, $m_{(u,t)} = le(u) - ed(t)$ (≥ 0) if tour t picks up an airplane after demand u, and $m_{(u,t)} = 0$ otherwise. Moreover, $m^u_{(p,q)} = -1$ if $p = u$, $m^u_{(p,q)} = 1$ if $q = u$, and $m^u_{(p,q)} = 0$ otherwise. If $u \in V_{bd}$, then $m_{(u,t)}$ and $m^u_{(p,q)}$ are defined similarly.

(45): time window constraints imply that if a tour t picks up an airplane, then this pickup must take place at a time after $ed(t)$ and before $ld(t)$. Suppose that $p \in V_{ad}$ and $q \in V_{bd}$. Then $m_{(p,q,t)} = -(ld(t) - ed(t))$ if tour t picks up an airplane after demand p and then drops it off before demand q, and $m_{(p,q,t)} = 0$ otherwise. In other cases of p and q, $m_{(p,q,t)}$ is defined similarly.

In the following, we will show that constraints (44) and (45) imply that the pickup time of any tour in an optimal schedule is feasible.

Theorem 4.1.1. *$z_{(p,q)}$ in problem (39) is well-defined*

Proof. Constraints (45) provide an upper bound on $z_{(p,q)}$ as the following:

$$z_{(p,q)} \leq \sum_{t:(p,q)\in t} (ld(t) - ed(t))x_t.$$

$(p,q) \in t$ means that the airplane of tour t is picked up from $p \in U$ and is dropped off to $q \in U$. Since $\sum_{t:(p,q)\in t} x_t \leq 1$ in any optimal solution, we have the following: if there exists an optimal tour t such that $(p,q) \in t$ and $x_t = 1$, then

$$0 \leq z_{(p,q)} \leq ld(t) - ed(t); \tag{46}$$

otherwise, $z_{(p,q)} = 0$. This ensures the meaning of variable $z_{(p,q)}$: $ed(t) + z_{(p,q)}$ is the actual pickup time of tour t, which can not be later than $ld(t)$.

Suppose that t^* is a tour in an optimal solution x^*, i.e. $x^*_{t^*} = 1$. WLOG, assume that t^* picks up its airplane after demand $p^* \in V_{ad}$ and drops it off before demand $q^* \in V_{bd}$. Consider constraint (44) for p^*. It follows from (43) that we have the following:

$$\sum_{t\in T,(p,p^*)\in t} (ee(t) - le(p^*))x^*_t + \sum_{p,p^*\in U,p\neq p^*} z_{(p,p^*)} + (le(p^*) - ed(t^*)) - z_{(p^*,q^*)} \leq 0$$

Note there must exist exact one tour that drops an airplane after p^* in x^*. Let t^1 be such a tour, and we have the following:

$$(ee(t^1) - le(p^*)) + z_{(p,p^*)} + (le(p^*) - ed(t^*)) - z_{(p^*,q^*)} \leq 0$$

i.e. $ee(t^1) + z_{(p,p^*)} \leq ed(t^*)) + z_{(p^*,q^*)}$. In other words, the pickup time $ed(t^*)) + z_{(p^*,q^*)}$ is later than the drop-off time $ee(t^1) + z_{(p,p^*)}$.

Moreover, note that $ee(p^*) \leq ee(t^1) \leq le(t^1) \leq le(p^*))$ and $ee(p^*) \leq ed(t^*) \leq ld(t^*) \leq le(p^*)$. Therefore, it follows from (46) that $ee(p^*) \leq ee(t^1) + z_{(p,p^*)} \leq le(p^*)$ and $ee(p^*) \leq ed(t^*) + z_{(p^*,q^*)} \leq le(p^*)$.

Furthermore, if no tours will pick up the airplane, then the summation in left hand side of (44) is non-positive, thus (44) still holds.

Other cases for $p^*, q^* \in U$ follow similarly, and we have that the feasibility of the actual drop-off time and pickup time of a tour is implied by constraints (44) and (45). □

Let us denote the MIP problem (39) by MIP(39) and denote by LP(39) its LP relaxation in which we require x and y to be non-negative instead of binary. Note that $x \leq 1$ and $y \leq 1$ are implied by constraints (41) and (42). Therefore, LP(39) is a valid relaxation.

Dual of the LP(39) is the following:

$$
\begin{aligned}
Max \quad & \sum_{t \in T} c_t x_t + \sum_{d \in \mathbb{D}} c_d y_d && && (47)\\
s.t. \quad & -\sum_{d \in t} \gamma_d - \sum_{c \in t} \gamma_c - \gamma_w + \gamma_v + (ld(t) - ed(t))\gamma_{(p,q)} - \gamma_p m_{(p,t)} - \gamma_q m_{(q,t)} && &&\\
& \leq c_t, && \forall\, t \in T, (w,v) \in t, (p,q) \in t && (48)\\
& -\gamma_d \leq c_d, && \forall\, d \in \mathbb{D} && (49)\\
& \gamma_p - \gamma_{(p,q)} - \gamma_q \leq 0, && \forall\, p, q \in U \text{ and } p \neq q && (50)\\
& \gamma(\text{ except for } \gamma_d) \geq 0 &&
\end{aligned}
$$

Let us denote the dual problem (47) by DLP(47). In constraints (48), $(w, v) \in t$ means that tour t picks up its airplane from $w \in A$ or $w \in W^g$, and drops it off at $v \in W^g$, and $(p, q) \in t$ means that tour t picks up its airplane from $p \in U$, and drops it off at $q \in U$.

In DLP(47), variable γ_d corresponds to (42), γ_c corresponds to (41), γ_w corresponds to (40) if $w \in A$ and corresponds to (43) if $w \in W^g$, γ_v corresponds to (43), $\gamma_{(p,q)}$ corresponds to (45), and γ_p and γ_q correspond to (44).

4.2 Column Generation Approach

Let us consider the following problem:

$$Min \quad \sum_{i \in T} c_i x_i \tag{51}$$

$$s.t. \quad \sum_{i \in T} a_i x_i \leq b \tag{52}$$

$$x_i \geq 0, \forall\ i \in T$$

If a simplex algorithm is used to solve LP (51), then we need to find a non-basic variable with the minimum reduced cost to enter the basis at each iteration, i.e. we calculate the following minimum explicitly:

$$Min\{c_i - y^T a_i : a_i \in A\}. \tag{53}$$

If this minimum is non-negative, then current solution x is optimal. Otherwise, if the minimum is attained at $j \in T$, then x_j will enter the basis.

However, if $|T|$ is huge or it is impossible to generate the whole constraint matrix $A = \{a_1, ..., a_{|T|}\}$ in advance, and if (53) can be solved implicitly, then we can apply similar ideas and do the followings:

step 1: in order to get dual variables, solve LP (51) that is generated based on a subset of columns, denoted as the *restricted master problem* (RMP);

step 2: replace y in (53) with optimal dual variables of the RMP and solve problem (53);

step 3: if the optimal objective value of problem (53) is non-negative, then current optimal solution to the RMP is also optimal to LP (51). Otherwise, add the column a_i^* that is optimal to (53) to the RMP and go to step 1.

The key ingredient in the above generic column generation algorithm is how to solve problem (53), which is also referred to as the *pricing subproblem*.

In our setting, we want to solve LP(39). So we start with a subset of tours and solve the RMP to get the optimal dual variables, which also satisfy constraints (49) and (50) in the DLP(47). Therefore, we only need to check whether the optimal dual variables of the RMP also satisfy constraints (48) in the DLP(47). In other words, the pricing subproblem becomes the following:

$$Min\{rc(t) : t \in T\}. \tag{54}$$

In (54), $rc(t)$, the reduced cost of tour t, is defined as the following:

$$rc(t) = c_t + \sum_{d \in t} \gamma_d + \sum_{c \in t} \gamma_c + \gamma_w - \gamma_v - (ld(t) - ed(t))\gamma_{(p,q)} + \gamma_p m_{(p,t)} + \gamma_q m_{(q,t)},$$

in which $w, v \in W$, $(w, v) \in t$, $p, q \in U$, $(p, q) \in t$, and γ is the optimal dual variables of the RMP. Therefore, the pricing subproblem (54) is to find a feasible tour with the minimum reduced cost.

It follows from the previous chapter that each feasible tour corresponds to a path in the duty network N_f, but a path in N_f may not corresponds to a feasible tour. Moreover, $rc(t)$ contains items that depend on the time window of tour t, which is not known unless the whole tour is known. Furthermore, 10-in-24 rule may need to be incorporated into the model in the future. Therefore, considering all the above, a shortest path algorithm applied to the N_f is not suitable to find a tour with the minimum reduced cost, and we will use a constrained shortest path algorithm instead.

In addition, we also the following methods to accelerate column generation process:

- generate a set of columns with negative reduced costs at each iteration, which is easy to do since our pricing subproblem is solved by a dynamic programming approach;

- stop column generation when upper bound and lower bound of the optimal cost are close enough, because of the well-known tailing-off effect of a column generation procedure;

- manage the added columns by either of the followings:
  - given a parameter n, we only keep n smallest columns in term of reduced cost;
  - given a parameter ϵ, we only keep columns with reduced cost less than ϵ.

  when the size of the RMP, i.e. number of columns already added, becomes huge;

A constrained shortest path problem is an extension of the shortest path problem with additional resource constraints. For example, the shortest path problem with time windows has the additional requirement that the time window constraint at each node of a path must be satisfied.

It follows from Theorem 3.2.5 that N_f is acyclic. Therefore, we will solve the pricing subproblem, using a label setting algorithm that contains the following two basic steps:

- path extension, i.e. given a path P to node n_1 and an arc from n_1 to n_2, how to check the feasibility and extend P to a path to n_2;
- dominance checking, i.e. given a set of paths to a node, how to identify paths that can not be extended to a minimum reduced cost path and discard them.

4.2.1 Lower Bound

Let us write the RMP of the LP(39) as the following:

$$Min \qquad \sum_{t\in T^s} c_t x_t + \sum_{d\in D} c_d y_d + \sum_{p,q\in U} 0 \cdot z_{(p,q)} \tag{55}$$

$$s.t. \qquad \sum_{t\in T^s} A_t^T x_t + By + Cz \leq b \tag{56}$$

$$x_t \geq 0, \forall\ t \in T^s, y \geq 0, z \geq 0$$

In (55), T^s is a subset of T. Let $T = \cup_{1 \leq i \leq n} T^i$ be a partition of T such that $\sum_{t \in T^i} x_t \leq 1$. In other words, each T^i is an independent subproblem of the pricing subproblem such that only at most one tour from T^i can be used in the optimal schedule.

Theorem 4.2.1. *Let z^s be the optimal objective value of RMP(55), let z^* be the optimal objective value of LP(39), and let r^i be the optimal objective value of the pricing subproblem (54) on tours T^i, given the optimal dual variables γ^s of RMP(55). Then we have the following:*

$$z^s + \sum_{1 \leq i \leq n} min\{r^i, 0\} \leq z^* \leq z^s$$

Proof. Let us consider the Lagrange dual of LP(39). Let $\gamma \geq 0$ be any Lagrange multipliers, and we have the following:

$$\begin{aligned} z^* &\geq min\{\sum_{t \in T} c_t x_t + \sum_{d \in D} c_d y_d + \sum_{p,q \in U} 0 \cdot z_{(p,q)} - \gamma^T (b - \sum_{t \in T} A_t^T x_t - By - Cz) \\ &\quad : x, y, z \geq 0\} \qquad (57) \\ &\geq min\{\sum_{t \in T} (c_t + \gamma^T A_t^T) x_t + \sum_{d \in D} (c_d + \gamma_d) y_d + \sum_{p,q \in U} (\gamma_q + \gamma_{(p,q)} - \gamma_p) z_{(p,q)} \\ &\quad - \gamma^T b : x, y, z \geq 0\} \qquad (58) \end{aligned}$$

Now we can substitute γ in (58) with γ^s. It follows from (49) that $c_d + \gamma_d^s \geq 0$. Since $y \geq 0$, we have that the term $\sum_{d \in D} (c_d + \gamma_d) y_d$ in (58) is zero in order to minimize (58), by setting y to be equal to 0. Similarly, it follows from (50) that $\gamma_q^s + \gamma_{(p,q)}^s - \gamma_p^s \geq 0$, thus $\sum_{p,q \in U} (\gamma_q + \gamma_{(p,q)} - \gamma_p) z_{(p,q)} = 0$ in order to minimize (58).

Therefore, (58) becomes the following;

$$
\begin{aligned}
(58) &= min\{\sum_{t\in T}(c_t + (\gamma^s)^T A_t)x_t : x \geq 0\} - (\gamma^s)^T b \\
&= min\{\sum_{1\leq i\leq n}\sum_{t\in T^i}(c_t + (\gamma^s)^T A_t)x_t : x \geq 0\} - (\gamma^s)^T b \quad (\text{since } T = \cup_{1\leq i\leq n} T^i) \\
&\geq min\{\sum_{1\leq i\leq n} r^i(\sum_{t\in T^i} x_t) : x \geq 0\} - (\gamma^s)^T b \quad (r^i \text{ is the min over } T^i) \\
&= \sum_{1\leq i\leq n} min\{r^i, 0\} - (\gamma^s)^T b \quad (\text{set } \sum_{t\in T^i} x_t \text{ to 0 or 1}) \\
&= \sum_{1\leq i\leq n} min\{r^i, 0\} + z^s \quad (\text{duality of the RMP})
\end{aligned}
$$

It follows that $z^* \geq z^s + \sum_{1\leq i\leq n} min\{r^i, 0\}$. Moreover, since RMP(55) is based on a subset of all feasible tours, we have that $z^s \geq z^*$, which completes the proof. □

4.2.2 Pricing Sub-problem

Note that each feasible tour corresponds to a feasible $n_s - N_{da} \cup N_{db}$ path in N_f, i.e., a path that originates at n_s and ends at a node in $N_{da} \cup N_{db}$. And a partial path is feasible if this partial path can be extended to a feasible $n_s - N_{da} \cup N_{db}$ path.

We will divide the set of all feasible tours into disjoint subsets, and each subset consists of all feasible tours ending at a node in $N_{da} \cup N_{db} \subseteq N_f$, for each fleet type f. Therefore, the pricing sub-problem is divided into a set of parallel problems, each of which corresponds to finding a minimum-reduced-cost feasible path in N_f that ends at d, for $d \in N_{da} \cup N_{db} \subset N_f$ and for $f \in F$.

Given $d \in N_{da} \cup N_{db}$, in order to solve the corresponding pricing sub-problem, which is a constrained shortest path problem, we will use a label-setting algorithm that starts at d, and goes backward to extend feasible partial paths. Note that N_f is acyclic.

More specifically, let $v \in N_{da} \cup N_{db} \subseteq N_f$ and let γ be optimal dual variables of the RMP(39). In the label-setting algorithm, we will use a label on a duty node

S to denote a partial path from S to v, and this label should contain all necessary information for the path extension and path dominance steps.

Let us consider the definition of the reduced cost of a feasible tour t:

$$rc(t) = c_t + \sum_{d \in t} \gamma_d + \sum_{c \in t} \gamma_c + \gamma_w - \gamma_v - (ld(t) - ed(t))\gamma_{(p,q)} + \gamma_p m_{(p,t)} + \gamma_q m_{(q,t)} \quad (59)$$

The key observation about the reduced cost (59) is that it can not be allocated to arcs and nodes in the corresponding path, i.e., it can not be written as a summation of cost on each arc or node in the path.

Let $P(S, v)$ be the partial path from duty node S to v. It follows that $P(S, v)$ must contribute the following to the reduced cost of any feasible path containing $P(S, v)$:

$$rc(P(S, v)) = c_{P(S,v)} + \sum_{d \in P(S,v)} \gamma_d - \gamma_v.$$

Although γ is given, other terms in (59), e.g., $(ld(t) - ed(t))$, $m_{(p,t)}$ and $m_{(q,t)}$, depend on the whole path, and can not be determined by only considering the partial path $P(S, v)$. However, these terms only depend on $ed(P(S, v))$, $ld(P(S, v))$ and $ece(P(S, v))$.

Recall that in section 3.3.3, we defined $[\alpha_v, \beta_v]$ for the crew end time of tours ending at v, and it implies that the crew end times have the same effects if they are before time α_v. Therefore, let $du(P(S, v))$ be the crew duration of $P(S, v)$. Note that since $P(S, v)$ is a feasible partial path, $ed(P(S, v)) \leq \beta_v - du(P(S, v))$. Moreover, we will update $ed(P(S, v))$ to be the following:

$$min\{ld(P(S, v)), max\{ed(P(S, v)), \alpha_v - du(P(S, v))\}\},$$

and update $ld(P(S, v))$ to be

$$min\{ld(P(S, v)), max\{ed(P(S, v)), \beta_v - du(P(S, v))\}\}.$$

Moreover, let $a(P(S, v)) = max\{ed(P(S, v)) + du(P(S, v)) - \alpha_v, 0\}$, and $a(P(S, v))$ denotes the crew end time of $P(S, v)$ comparing to α_v when $P(S, v)$ starts at time $ed(P(S, v))$.

It follows that, in order to calculate the total reduced cost of a feasible path and consider the dominance of partial paths to the same duty node, we need to keep $ed(P(S,v))$, $ld(P(S,v))$ and $a(P(S,v))$ for each partial path $P(S,v)$.

Furthermore, let a be the arc after duty node S in $P(S,v)$. In order to extend $P(S,v)$ feasibly, we need to consider the total flight time and duty time of the duty induced by S and a.

Definition 4.2.1. *Let $v \in N_{da} \cup N_{db}$. A valid* label *on duty node S that corresponds to a partial path $P = P(S,v)$ is defined by $l(S) = \{e_S, l_S, a_S, b_S, d_S, c_S\}$, in which $e_S = ed(P)$, $l_S = ld(P)$, $a_S = a(P)$, and $c_S = rc(P)$. If S is a crew-duty, then b_S is the total flight time of duty $S \cup a$ and d_S is the crew end time of $S \cup a$ when P starts at time $e_S = ed(P)$. Otherwise, $b_S = 0$ and $d_S = 0$.*

The following proposition is used to eliminate dominated labels in the path extension steps.

Proposition 4.2.1. *Given $v \in N_{da} \cup N_{db}$ and a duty node $S \in N_f$, let $l^1(S) = \{e^1, l^1, a^1, b^1, d^1, c^1\}$ and $l^2(S) = \{e^2, l^2, a^2, b^2, d^2, c^2\}$ be two labels on duty S. $l^1(S)$ dominates $l^2(S)$ if all the followings hold:*

(a) $l^1 \geq l^2$;

(b) $a^1 \leq a^2 + \min(0, e^1 - e^2)$;

(c) $b^1 \leq b^2$;

(d) $d^1 \leq d^2 + \min(0, e^1 - e^2)$;

(e) $c^1 \leq c^2$.

4.3 Computational Results

In this section, we will show the computational results of our model. Three types of computations are tested and presented:

1. Compare our model with the model used in practice to see how our model performs on average;

2. Consider various assumptions/strategies used in practice, and study their impact on the solution quality and running time;

3. Test our model in a rolling horizon procedure over a longer planning to see whether our solution is consistent.

Our model is implemented in C and CPlex 11 Callable Library, and is run on a desktop with P4 CPU 2.4GHz and 2 Gigabytes of RAM. We obtain input data from the actual data of a major fractional airline company, which includes almost every aspect of the actual operations. Actual data also includes the actual schedules and actual crew activities, e.g. legs with assigned airplanes and pilots, and each pilot's activity on each day (on-duty, off-duty, in-training etc.).

Given a planning window $[t_a, t_b]$, our input data is obtained from actual data as follows:

- Demands consist of all customer-requested flights with requested departure time in $[t_a, t_b]$, and all maintenance requests and appointments with start time in $[t_a, t_b]$.

- From the actual schedule and for each airplane, get the last leg with departure time no later than t_a, and set available time and available airport of this airplane to be the arrival time and arrival airport of this leg, respectively.

- Set a time t_c such that $t_c < t_a$, and t_c implies that crew related activities, i.e., rest, travel, flight etc., must start after time t_c.

- From the actual schedule and for each pilot i, get the last leg with departure time no later than t_a. Denote this last leg by g_i, and set available time and available airport of pilot i to be the arrival time and arrival airport of leg g_i,

respectively. Moreover, if pilot i is available before time t_c and is on-duty when available, then let a pilot rest in the following cases:

  - There is enough time to rest after being available and before time t_c ;
  - Duty time exceeds the maximum when available at time t_c ;
  - In the actual schedule and after leg g_i, pilot i is assigned to leg h in $[t_a, t_b]$, and duty time or block time exceeds if this pilot doesn't rest after leg g and before leg h, and there is not enough time to rest after time t_c and before leg h .

  Note that if a pilot is not assigned to any legs in the actual schedule, but available according to crew schedule, then we set this pilot to be available at his/her base airport.

- Crew pairs (a pair of pilots) are obtained in another module, which is based on the actual schedule of $[t_a, t_b]$ and solutions of a bipartite matching problem.

Note that when our model is used in a rolling horizon procedure, the above ideas are also used to update the data after each rolling period.

4.3.1 Average performance

In the section, we will test our model on a set of 21 two-day instances from actual data, including three peak day instances. Each peak day instance contains a high-level peak day, on which each demand is set to have a time window of six hours. Number of demands ranges from 127 to 204 with an average 171. For the non-peak day instances, average percentage of number of demands that have non-trivial time windows is 9.4%, and this percentage is 49.6% for peak day instances. Note that maintenance requests and appointments have no time window in these instances.

We will compare the solutions of our model with the solutions of the model used in practice, and show the average performance and improvement of our model.

Instance	Total	Demand	Repo	Travel	Charter	Penalty	Time	Opt Gap	#Vectors	#Demands
1	389,515.01	285,717.37	94,280.18	9,517.46	0.00	0.00	188	0.28%	18	127
2	616,404.79	348,651.30	143,957.85	9,854.99	73,940.65	40,000.00	268	0.06%	22	151
3	491,930.78	335,498.03	125,733.90	16,966.84	13,732.00	0.00	293	0.23%	16	148
4	485,520.78	288,651.60	142,507.47	10,437.32	43,924.40	0.00	410	0.01%	7	150
5	473,956.06	331,515.37	133,661.00	8,779.70	0.00	0.00	428	0.00%	15	160
6	642,175.51	423,545.42	178,770.97	19,859.13	0.00	20,000.00	485	0.12%	18	180
7	615,107.51	389,202.82	193,647.50	18,403.19	13,854.00	0.00	498	0.34%	16	172
8	568,772.56	379,788.38	158,747.05	16,383.13	13,854.00	0.00	584	0.72%	15	172
9	594,212.37	358,921.40	171,369.78	13,442.39	30,478.80	20,000.00	595	0.02%	17	190
10	575,411.39	353,847.38	186,679.03	21,152.97	13,732.00	0.00	662	0.21%	25	181
11	511,313.10	340,068.77	153,499.40	17,744.93	0.00	0.00	885	0.20%	8	173
12	638,002.41	406,916.98	214,120.78	16,964.64	0.00	0.00	1,067	0.07%	18	192
13	671,166.17	410,518.28	162,761.75	22,457.19	75,428.95	0.00	1,075	0.15%	13	195
14	599,831.14	417,796.10	155,000.83	27,034.20	0.00	0.00	1,098	0.22%	18	187
15	637,721.96	387,466.22	183,615.38	24,745.11	41,895.25	0.00	1,123	0.08%	16	192
16	607,058.33	412,643.85	153,451.35	20,963.13	0.00	20,000.00	1,156	0.07%	11	183
17	596,155.26	356,327.57	176,487.97	28,912.57	34,427.15	0.00	1,223	0.31%	18	184
18	609,609.41	401,343.45	173,065.72	21,346.24	13,854.00	0.00	1,280	0.04%	22	204
19	391,840.01	286,153.60	98,011.93	7,674.48	0.00	0.00	2,589	0.76%	82	144
20	411,759.95	301,239.05	93,681.12	16,839.78	0.00	0.00	3,321	0.59%	69	147
21	533,884.26	397,758.33	119,906.30	16,219.63	0.00	0.00	4,571	0.20%	80	178

Figure 7: Results of our model on 21 2-day instances

The results of our model are listed in the table in Figure 7. First column is the instance number, sorted in the increasing order of the running time. The second column through the seventh column represent total cost, cost of all customer requested flights that are not charter, cost of all repositioning flights, cost of all crew travels, cost of chartered flights, and penalties for uncovered maintenance request and appointments, respectively. "Time" column represents the running time in seconds. "Opt Gap" represents relative optimality gap. "#vector" column represents number of demands that have time windows in the instance. "#demands" represents total number of demands in the instance, including maintenance and appointments.

A first look at the table in Figure 7 reveals that the optimality gaps are small, with range $[0, 0.76\%]$ and average 0.22%, and running time has a wider range of $[188, 4573]$. The table in Figure 8 contains the results of the model used in practice, and columns have the same meaning as in Figure 7.

Figure 9 plots the percentage of cost reduction of the solution of our model over the solution of the model used in practice, and the percentage of reduction of the lower

Instance	Total	Demand	Repo	Travel	Charter	Penalty	Time
1	392,883.00	286,011.80	97,605.05	9,266.15	0.00	0.00	47
2	619,052.45	348,200.48	146,347.23	10,564.08	73,940.65	40,000.00	98
3	494,647.53	336,047.65	140,023.98	18,575.90	0.00	0.00	108
4	490,977.80	289,905.42	147,074.67	10,073.32	43,924.40	0.00	137
5	477,459.27	331,723.92	135,885.52	9,849.83	0.00	0.00	123
6	669,984.96	424,360.68	188,487.12	17,137.16	0.00	40,000.00	119
7	640,779.59	386,353.62	193,533.47	15,873.56	45,018.95	0.00	88
8	578,469.29	379,990.48	167,807.08	13,777.72	16,894.00	0.00	138
9	608,210.08	360,231.80	178,653.67	18,845.82	30,478.80	20,000.00	160
10	617,820.12	352,669.45	195,490.95	22,073.72	27,586.00	20,000.00	236
11	534,377.33	339,103.13	164,233.15	17,187.05	13,854.00	0.00	183
12	651,446.63	408,158.30	228,197.33	15,091.00	0.00	0.00	248
13	695,317.65	408,988.10	171,965.17	27,238.03	87,126.35	0.00	297
14	607,664.80	418,300.10	163,191.82	26,172.88	0.00	0.00	201
15	652,050.05	387,067.20	179,155.28	23,932.32	41,895.25	20,000.00	209
16	650,161.78	411,169.33	156,003.02	22,989.43	0.00	60,000.00	265
17	647,447.86	356,263.78	175,531.02	27,371.91	48,281.15	40,000.00	292
18	630,018.32	398,085.47	182,674.90	21,203.60	28,054.35	0.00	275
19	411,041.12	285,520.88	118,510.77	7,009.47	0.00	0.00	145
20	440,088.76	301,153.75	109,576.58	15,504.43	13,854.00	0.00	199
21	558,079.40	399,146.95	143,228.15	15,704.30	0.00	0.00	178

Figure 8: Results of original model on 21 2-day instances

bound of our model as reference. Figure 9 shows that our model reduces the total cost on each instance, and suggests a trend that the average reduction is increasing as the instance number increases, i.e., as the running time increases. Averaging over 21 instances, total cost reduces 3.3%, repositioning cost reduces 5.43%, and charter cost reduces 21.62%.

Figure 10 contains two pie charts that show the proportions of various costs in the total cost for both models. The left chart (case 0) corresponds to the model used in practice. Figure 10 shows that the proportions of both models are very similar. Comparing to the solution of the model used in practice, each type of cost in our solution reduces, except for the cost of demands, which is most likely due to the upgrading cost and less number of charter flights in our solution.

Figure 11 plots number of demands and running time of our model for each instance. The rightmost three instances are peak day instances, therefore their running times are significantly longer, although they have moderate number of demands. For

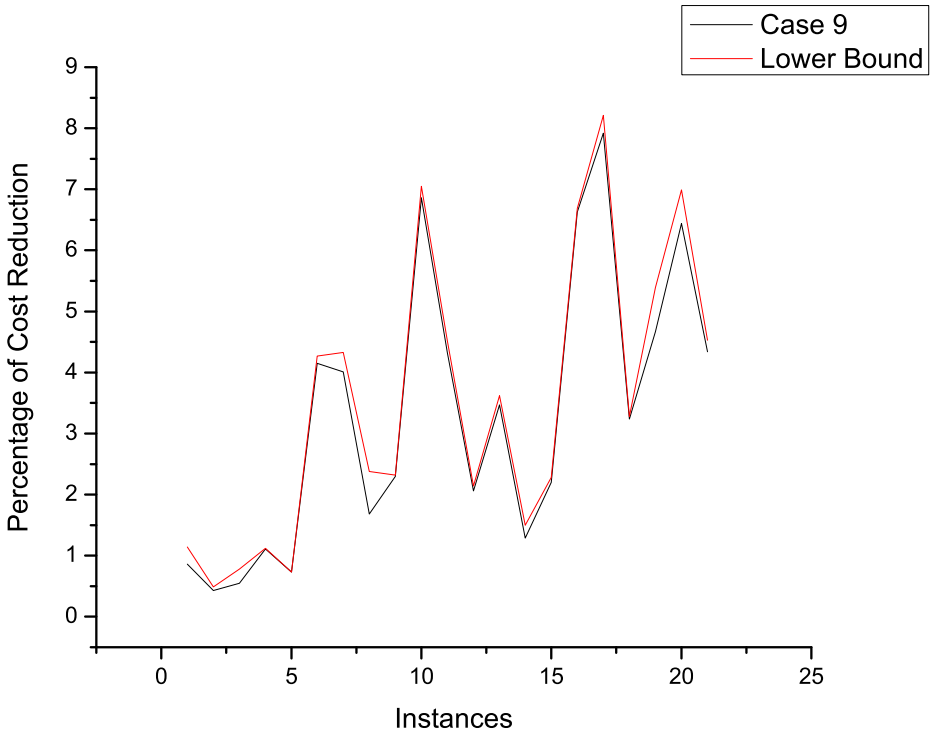


Figure 9: Comparison of total costs on 21 instances

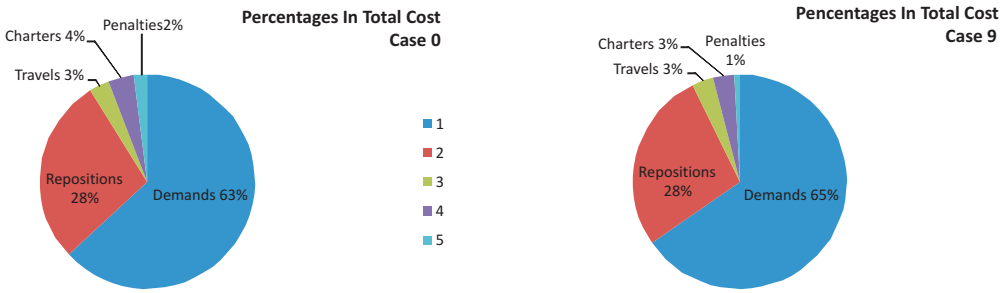


Figure 10: Proportions in total cost

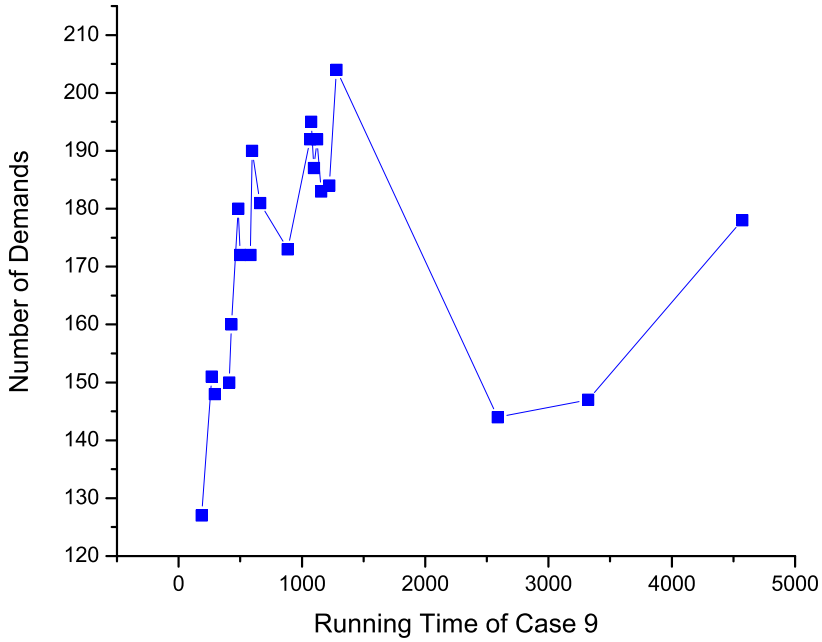


Figure 11: Number of demands and running time

all the non peak instances, Figure 11 shows the trend that running time increases as number of demands increases.

4.3.2 Strategies/assumptions

In the model used in practice, some assumptions or strategies are used to either reduce size of the model or to make the particular implementation work. We will list and discuss some of them, and study their effects in the solution quality and running time.

These assumptions are listed as follows:

1. **No duty across days**. Planning window is divided by days, e.g., start time of a day is 9AM GMT. If the requested departure time of a demand is within day i, then this demand is considered as from day i. The assumption is that a duty must consist of demands from the same day, except for a special case in which the last demand of a duty is a maintenance request from later days. It follows that this assumption limits feasibility of a duty, i.e., a feasible duty may

be considered as infeasible by the model.

2. **No repositioning only tours**. A tour can not consist of only a single repositioning leg. This assumption limits feasibility of a tour.

3. **Rest across day**. Rest period of a pilot must be across days, e.g., must contain 9AM GMT. This assumption limits crew rest between duties. In other words, it limits the feasibility of arcs between duties.

4. **Repositioning cutoff**. A time interval is specified, and if the last leg of a duty is a repositioning leg, then this leg must reposition to a demand whose requested departure time is inside this interval. Note that duties with a repositioning leg as the last leg are enumerated in the model used in practice. While in our model, only duties that both start and end with a demand are enumerated, and repositioning legs before or after a duty are contained in the arcs. Therefore, this assumption also limits the feasibility of arcs between duties.

5. **Crew no-crew duty**. Both crew start and crew end time of a no-crew duty are set to be equal to the start time of this duty. Therefore, if a pilot rests before a no-crew duty, then this pilot must finish resting before start time of this duty. But since a maintenance request or appointment does not require crews, if there is a non-trivial leg after this no-crew duty, then this pilot can rest until start time of next non-trivial leg.

   Similarly, if a pilot rests after a no-crew duty, then this pilot must start resting after start time of this duty. But if there is a non-trivial leg before this no-crew duty, then this pilot can rest right after end time of the previous non-trivial leg, which may be much earlier than start time of the no-crew duty. Therefore, this assumption limits the feasibility of arcs between duties.

6. **No duty time window**. Once a duty is generated, timings of all legs in this

Groups	Constraints	Assumptions
A	duty and tour feasibility	1,2
B	arc feasibility	3,4,5
C	duty time window	6
D	tour time window	7
E	maintenance time window	8

Table 1: Assumption Groups

duty are fixed. Therefore, a duty comes with a feasible timing that does not necessarily minimize the duration or duty time, and this timing is not allowed to change during the optimization. This assumption limits the feasibility of duties and arcs.

7. **No tour time window**. More specifically, if the last leg of a duty is a repositioning leg to a demand d, then this leg must arrive before the earliest departure time of d. And if an airplane is dropped off after a demand d, then this airplane is available at the latest arrival time of d. This assumption limits the feasibility of arcs and aircraft connections between tours.

8. **No maintenance time window**. Maintenance requests are not allowed to have a time window.

It follows that assumptions 1 and 2 are related to the duty and tour feasibilities, assumptions 3,4 and 5 are related to arc feasibilities. Therefore, size of the model reduces with these two sets of assumptions. Assumptions 6, 7, and 8 are related to time windows, and they limit feasibilities of duties, arcs and tours. Therefore, we group these assumptions into 5 groups as in table (1). Let $U = A \cup B \cup C \cup D \cup E$, i.e., the set of assumptions 1 through 8.

Note that our model does not have any of these assumptions. In order to study the impact of these assumptions, we will add appropriate constraints in our solution approach to reproduce the exact assumption in the model used in practice. However,

Cases	Assumptions Not Added
1	$\emptyset$
2	A
3	B
4	C
5	D
6	$A \cup B$
7	$A \cup B \cup D$
8	$A \cup B \cup C$
9	$A \cup B \cup C \cup D$
10	$A \cup B \cup C \cup D \cup E$

Table 2: Tested Cases For Assumptions

note that our model and the model used in practice are two completely different implementations. So an exact reproduction of an assumption may not be possible. But with close simulations, we shall still be able to see some of the impact of an assumption.

For example, assumption 6 (no duty time window) is added by first generating all feasible duties with time windows, then getting a fixed timing for each duty that is as close to what the model used in practice would produce as possible. Therefore, with this assumption, duty feasibility is not affected, but arc and tour feasibility may be affected, and we want to see the difference in the solutions.

We will test our model for all the cases in Table 2. Note that case 1 means that we add all assumptions 1 though 8 to our model. Therefore, case 2 though case 5 deal with the marginal improvement of removing each individual assumption group from case 1. Case 7, 8 and 9 are related to removing either or both time window assumptions from case 6. Moreover, in case 10, we add a two-hour time window to each maintenance request in the input data, and the purpose is to see the improvement comparing to case 9.

Tables in Figure 12 and Figure 13 contain the results of six instances. Each table corresponds to an instance, and each instance is tested for all cases in Table 2. Tables

	Total	Demand	Repo	Travel	Charter	Penalty	Time	Bound
0	650,161.78	411,169.33	156,003.02	22,989.43	0.00	60,000.00	265	
1	616,110.10	408,117.28	142,842.65	26,331.82	18,818.35	20,000.00	397	289.37
2	616,110.10	408,117.28	142,842.65	26,331.82	18,818.35	20,000.00	410	289.36
3	613,337.53	408,855.57	143,360.78	22,302.83	18,818.35	20,000.00	780	1,123.41
4	613,516.34	408,221.88	144,223.53	22,252.57	18,818.35	20,000.00	451	568.77
5	616,110.10	408,117.28	142,842.65	26,331.82	18,818.35	20,000.00	437	290.10
6	613,337.53	408,855.57	143,360.78	22,302.83	18,818.35	20,000.00	1001	1,123.41
7	607,760.69	412,708.78	152,785.53	22,266.37	0.00	20,000.00	1178	545.93
8	612,748.37	407,743.85	142,311.85	23,874.32	18,818.35	20,000.00	952	1,387.60
9	607,058.33	412,643.85	153,451.35	20,963.13	0.00	20,000.00	1156	421.04
10	601,455.24	412,258.50	146,840.60	22,356.14	0.00	20,000.00	1054	894.65

(a) Instance 1, Non-peak day

	Total	Demand	Repo	Travel	Charter	Penalty	Time	Bound
0	640,779.59	386,353.62	193,533.47	15,873.56	45,018.95	0.00	88	
1	622,958.89	390,309.87	200,091.02	18,704.01	13,854.00	0.00	119	532.87
2	622,834.46	390,347.38	199,929.07	18,704.01	13,854.00	0.00	114	408.43
3	620,335.77	389,102.67	196,358.03	21,021.07	13,854.00	0.00	177	305.84
4	623,706.64	390,431.93	201,317.93	18,102.77	13,854.00	0.00	154	2,127.47
5	622,958.89	390,309.87	200,091.02	18,704.01	13,854.00	0.00	159	810.61
6	620,335.77	389,102.67	196,358.03	21,021.07	13,854.00	0.00	192	305.83
7	615,890.40	393,156.45	204,108.50	18,625.45	0.00	0.00	367	2,563.16
8	619,782.46	389,102.67	195,827.12	20,998.68	13,854.00	0.00	250	562.68
9	615,107.51	389,202.82	193,647.50	18,403.19	13,854.00	0.00	498	2,079.67
10	599,144.05	392,735.15	189,103.47	17,305.44	0.00	0.00	475	479.22

(b) Instance 2, Non-peak day

	Total	Demand	Repo	Travel	Charter	Penalty	Time	Bound
0	477,459.27	331,723.92	135,885.52	9,849.83	0.00	0.00	123	
1	476,260.01	331,659.28	134,874.23	9,726.50	0.00	0.00	176	95.10
2	476,260.01	331,659.28	134,874.23	9,726.50	0.00	0.00	164	118.93
3	475,948.75	331,484.23	134,875.67	9,588.85	0.00	0.00	373	39.21
4	475,011.75	332,223.60	133,506.15	9,282.00	0.00	0.00	212	40.53
5	475,978.47	331,857.57	133,828.33	10,292.57	0.00	0.00	172	22.62
6	475,781.99	331,484.23	134,875.67	9,422.09	0.00	0.00	453	5.56
7	474,925.30	331,694.07	133,958.70	9,272.54	0.00	0.00	370	36.14
8	474,387.19	331,678.87	133,309.57	9,398.76	0.00	0.00	281	0.00
9	473,956.06	331,515.37	133,661.00	8,779.70	0.00	0.00	428	-0.01
10	460,269.89	331,454.77	119,219.05	9,596.07	0.00	0.00	556	299.45

(c) Instance 3, Non-peak day

	Total	Demand	Repo	Travel	Charter	Penalty	Time	Bound
0	534,377.33	339,103.13	164,233.15	17,187.05	13,854.00	0.00	183	
1	514,765.30	340,913.95	152,858.58	20,992.77	0.00	0.00	287	559.20
2	514,765.30	340,913.95	152,858.58	20,992.77	0.00	0.00	250	560.05
3	512,633.35	340,178.38	151,181.83	21,273.13	0.00	0.00	734	1,022.41
4	514,759.39	339,826.08	153,618.67	21,314.64	0.00	0.00	383	647.13
5	514,742.13	339,826.08	153,595.58	21,320.46	0.00	0.00	410	628.99
6	512,633.35	340,178.38	151,181.83	21,273.13	0.00	0.00	495	1,106.27
7	511,313.10	340,068.77	153,499.40	17,744.93	0.00	0.00	858	1,031.74
8	512,633.35	340,178.38	151,181.83	21,273.13	0.00	0.00	713	1,175.33
9	511,313.10	340,068.77	153,499.40	17,744.93	0.00	0.00	885	1,031.74
10	510,138.55	339,975.67	150,618.62	19,544.26	0.00	0.00	977	882.98

(d) Instance 4, Non-peak day

Figure 12: Non-peak day instances

	Total	Demand	Repo	Travel	Charter	Penalty	Time	Bound
0	411,041.12	285,520.88	118,510.77	7,009.47	0.00	0.00	145	
1	406,695.48	284,958.68	112,497.52	9,239.28	0.00	0.00	217	0.00
2	406,695.48	284,958.68	112,497.52	9,239.28	0.00	0.00	227	0.00
3	406,695.48	284,958.68	112,497.52	9,239.28	0.00	0.00	351	2.90
4	393,487.83	285,015.70	99,339.13	9,132.99	0.00	0.00	415	272.63
5	406,662.66	284,958.68	112,497.52	9,206.46	0.00	0.00	248	44.05
6	406,695.48	284,958.68	112,497.52	9,239.28	0.00	0.00	333	0.29
7	405,247.88	285,463.90	111,379.00	8,404.98	0.00	0.00	911	1,012.12
8	393,533.80	285,043.93	99,980.53	8,509.34	0.00	0.00	468	421.40
9	391,840.01	286,153.60	98,011.93	7,674.48	0.00	0.00	2,589	2,972.08
10	390,379.16	286,905.20	97,241.47	6,232.50	0.00	0.00	2,409	3,077.01

(a) Instance 5, Peak day

	Total	Demand	Repo	Travel	Charter	Penalty	Time	Bound
0	440,088.76	301,153.75	109,576.58	15,504.43	13,854.00	0.00	199	
1	433,913.00	302,294.55	112,340.30	19,278.15	0.00	0.00	282	3,311.67
2	434,112.96	303,147.43	112,318.73	18,646.80	0.00	0.00	265	3,922.06
3	425,185.66	301,209.67	106,627.72	17,348.28	0.00	0.00	522	2,508.97
4	422,363.65	300,815.33	103,004.35	18,543.97	0.00	0.00	418	535.24
5	421,914.91	300,523.60	100,048.30	21,343.01	0.00	0.00	341	336.86
6	424,245.25	301,243.18	108,129.52	14,872.55	0.00	0.00	526	2,509.60
7	419,034.13	300,826.93	100,932.83	17,274.37	0.00	0.00	1,252	1,137.75
8	416,963.97	301,408.57	99,890.25	15,665.15	0.00	0.00	738	675.19
9	411,759.95	301,239.05	93,681.12	16,839.78	0.00	0.00	3,321	2,421.48
10	410,831.00	301,053.03	93,500.03	16,277.93	0.00	0.00	3,084	1,920.95

(b) Instance 6, Peak day

Figure 13: Peak day instances

in Figure 12 correspond to four typical non-peak day instances, and tables in Figure 13 correspond to two instances that contain a peak day, during which a six hour time window is added to each demand.

Let us consider a table in Figure 12. Each row corresponds to a case with the same case number as in Table 2, and the the first row with row number 0 corresponds to the model that is used in practice. Columns correspond to total cost, cost of all customer requested demands that are not chartered, repositioning cost, travel cost, charter cost, penalty for uncovered maintenance requests or appointments, running time and the last column is the absolute optimality gap, i.e. total cost subtracted by the lower bound.

Figure 14 compares the total costs of case 0, 1, and 9. The x-axis is instance number, and the y-axis is the percentage of total cost reduction over case 0. Figure 14 shows the consistent improvement of case 1 over case 0, and shows that case 9 further improves case 1, especially on peak days. Comparison of case 9 and 1 implies

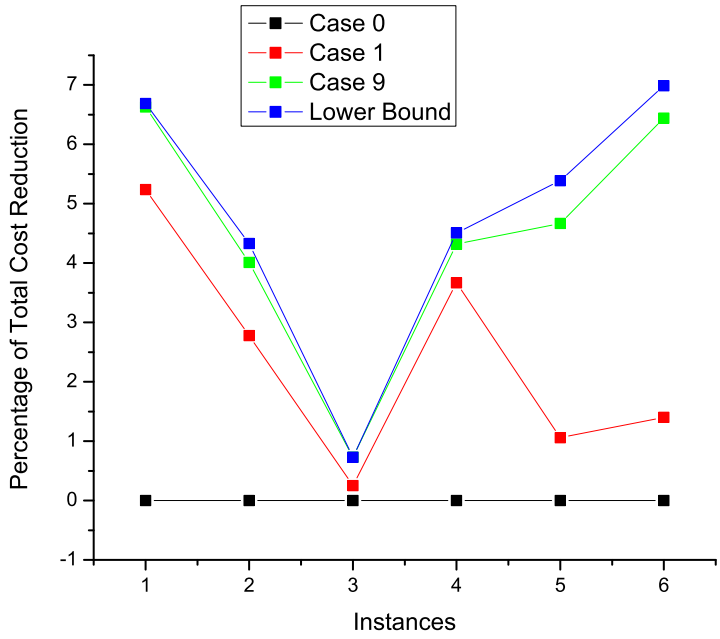


Figure 14: Total cost of case 0, 1, and 9

that big improvements can be achieved by exploring more feasible solutions, i.e., removing assumptions 1 through 8.

Figure 15 compares the travel costs to pick up airplanes in case 0, 1, and 9. The x-axis is instance number, and the y-axis is the travel cost subtracted by the travel cost of case 0. Figure 15 shows that travel cost in case 1 is greater than in case 0 in each instance. Note that we added all assumptions to our model to form case 1, but optimization of pickup arcs is still contained in case 1. Figure 15 suggests that the pickup arc optimization in case 1 allows the model to use more crews, thus increase the travel cost.

This observation is also suggested by Figure 16. The x-axis is instance number and the y-axis is number of tours. Figure 16 shows that case 1 has more tours than case 0 in every instance. Note that each crew can only be assigned to one tour. Therefore, case 1 uses more crews than case 0.

Figure 17 plots the running times in case 0, 1, and 9 of each instance. It shows that

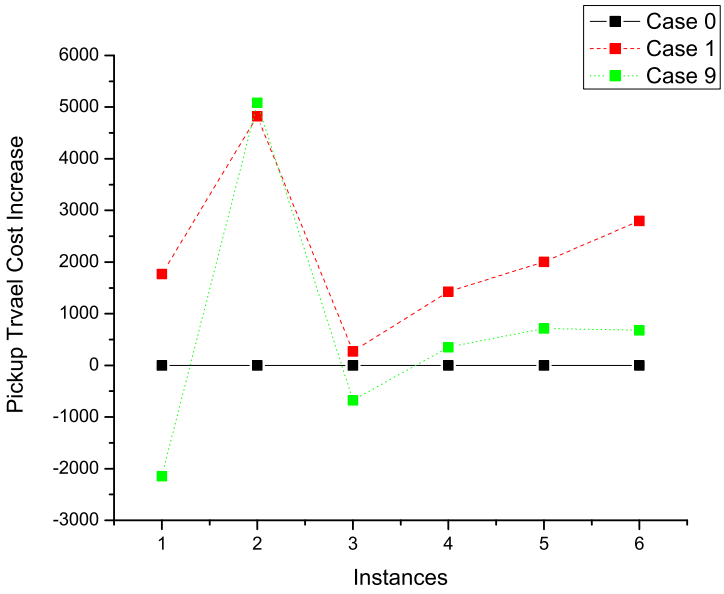


Figure 15: Travel cost of case 0, 1, and 9

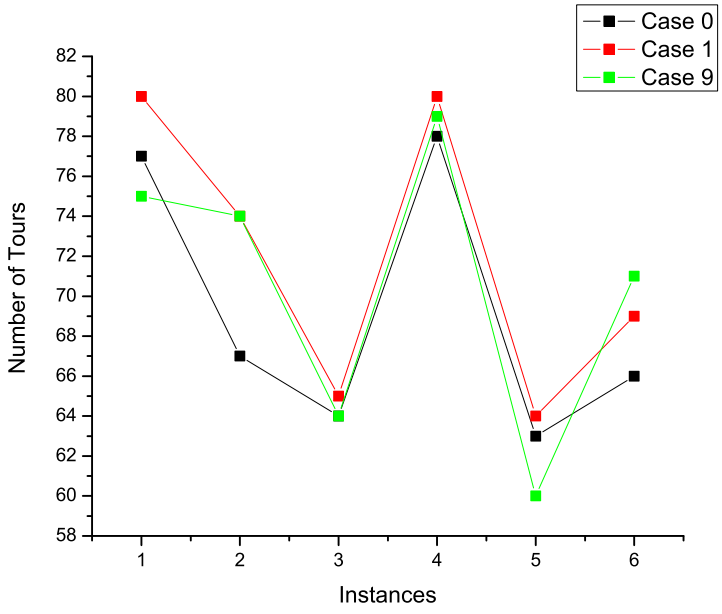


Figure 16: Number of tours in case 0, 1, and 9

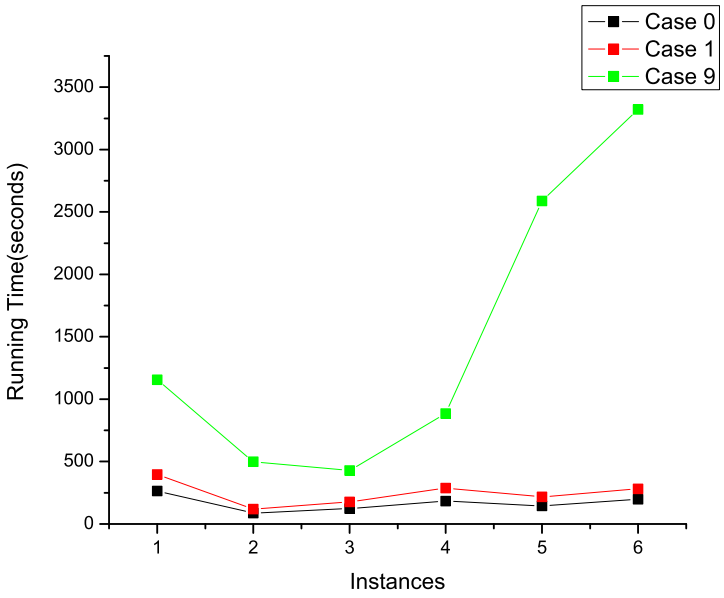


Figure 17: Running time of case 0, 1, and 9

running times of case 1 are very close to case 0, even on peak days, and running times of case 9 are much larger, especially on peak days. It follows that when running time has higher priority than solution quality, we can add all assumptions to our model, and use case 1, which will run in time comparable to the model used in practice and return better solutions.

Comparing to case 1, case 2 has very minimal improvement, and only reduces 0.04% in instance 6. Figure 18 compares total cost reduction of case 3, 4, and 5 over case 1. The x-axis is instance number, and the y-axis is the percentage of total cost reduction over case 0, subtracted by the total cost reduction of case 1 over case 0. Case 5, which removes the tour time window assumption from case 1, has minimal improvement over case 1, except for a large improvement in instance 6.

It follows that if we are only allowed to remover one assumption group from case 1, then Group A and D are not good choices. However, this does not mean that they will not help to have bigger improvements when combined with other groups of

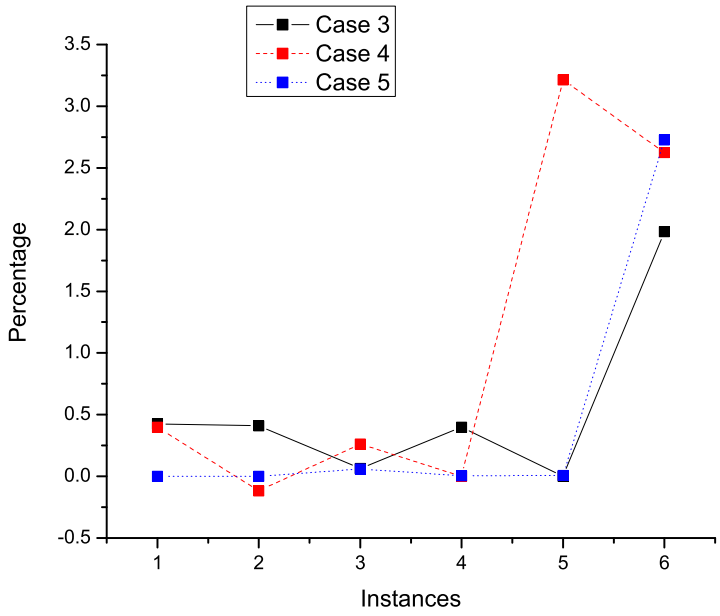


Figure 18: Total cost of case 3, 4, and 5

assumptions.

Figure 19 compares total costs of case 7, 8, and 9, i.e., improvements after removing tour time window assumption, or duty time window assumption, or both. The x-axis is instance number, and the y-axis is the percentage of total cost reduction over case 6. Figure 19 shows that total cost of case 7 is very close to total cost of case 9 on non-peak days, and total cost of case 8 is closer to total cost of case 9 on peak days.

Figure 20 compares total costs of case 9 and 10 and lower bound for case 10. The x-axis is instance number, and the y-axis is the percentage of total cost reduction over case 0. Figure 20 shows that adding time windows to maintenance may have big improvement in some instances.

4.3.3 Rolling Horizon

In this section, we will test case 0, 9, and 10, i.e., model used in practice, our model and our model with adding 2-hour time windows to maintenance, on a set of three

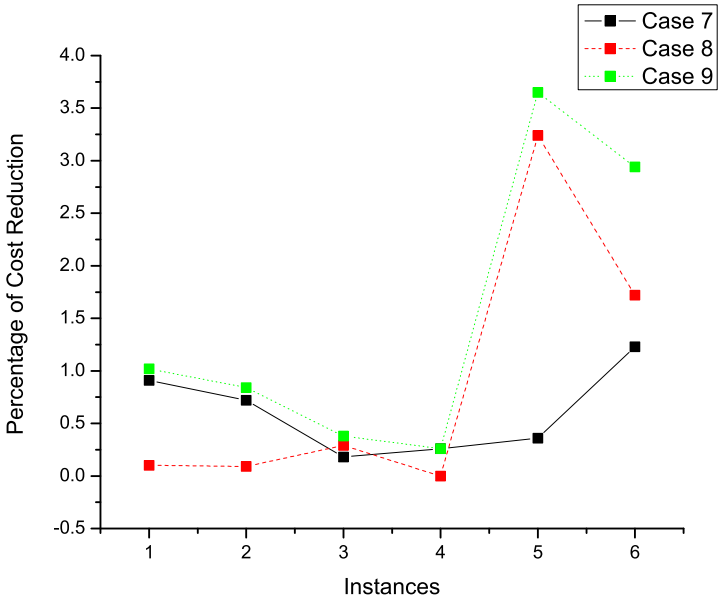


Figure 19: Total cost of case 7, 8, and 9

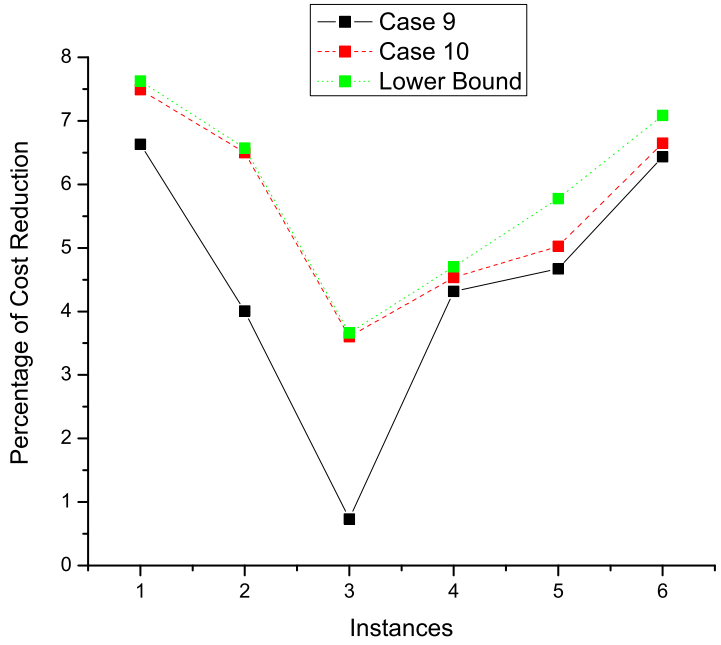


Figure 20: Total cost of case 9 and 10

Instance 1	Total	Demand	Repo	Travel	Charter	Penalty
Case 0	1,939,231.51	1,204,326.17	504,689.15	132,354.25	57,861.95	40,000.00
Case 9	1,872,294.95	1,217,015.55	516,573.45	98,705.95	0.00	40,000.00
Case 10	1,832,266.04	1,215,244.40	474,424.67	102,596.98	0.00	40,000.00

Instance 2	Total	Demand	Repo	Travel	Charter	Penalty
Case 0	1,757,864.93	1,087,734.93	458,691.43	147,080.57	44,358.00	20,000.00
Case 9	1,686,896.22	1,087,351.50	439,013.97	116,172.75	44,358.00	0.00
Case 10	1,679,758.15	1,086,496.50	435,636.87	113,266.78	44,358.00	0.00

Instance 3	Total	Demand	Repo	Travel	Charter	Penalty
Case 0	1,766,963.40	1,141,050.40	495,225.70	110,687.30	0.00	20,000.00
Case 9	1,725,717.46	1,142,071.65	495,121.95	88,523.86	0.00	0.00
Case 10	1,719,128.25	1,139,503.25	497,340.42	82,284.58	0.00	0.00

Figure 21: 7-day rolling horizon tests

7-day instances in a rolling horizon procedure. Rolling period is a single day, and the horizon at the start of each rolling period consists of two days. The results are shown in the tables of Figure 21.

It follows from Figure 21 that the six-day total cost of case 9 reduces 3.27% from case 0, averaging over three instances, and case 10 further reduces 0.98% from case 9. Note that the travel cost also reduces from case 0.

Figure 22 shows the cost on each day of the rolling horizon procedure for each instance. It follows that the daily cost of case 9 reduces 2.69% from case 0, averaging over 18 days, and daily cost of case 10 reduces 4.05% from case 0. Figure 23 plots the daily cost for each day of each instance.

Note that when considering daily cost in a rolling horizon procedure, case 9 or 10 has larger cost on some days. In other words, case 9 or 10 improves on some days, but does worse on other days. So one natural question is whether this observation is because of pure randomness.

To answer such a question, let us consider case 0 and 9, and do a p-value test with the following null hypothesis: whether case 9 improves over case 0 on a day in a rolling horizon procedure is random, i.e., probability that case 9 improves is 0.5. Figure 22 shows that case 9 improves over case 0 on 14 out of 18 days. Two-sided

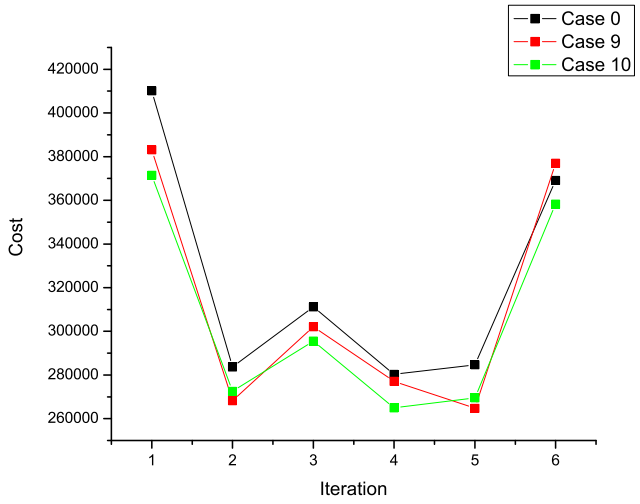
Instance 1	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6
Case 0	410,180.52	283,662.89	311,229.25	280,318.53	284,694.83	369,145.49
Case 9	383,145.23	268,217.23	302,122.21	277,114.29	264,739.07	376,956.92
Case 10	371,469.14	272,519.75	295,494.36	264,964.53	269,521.53	358,296.74

Instance 2	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6
Case 0	411,027.49	139,943.26	268,333.59	363,589.93	330,088.13	244,882.54
Case 9	408,187.91	149,173.55	251,215.43	331,652.63	311,134.79	235,531.89
Case 10	408,130.00	130,053.27	260,863.78	337,688.23	317,573.17	225,449.69

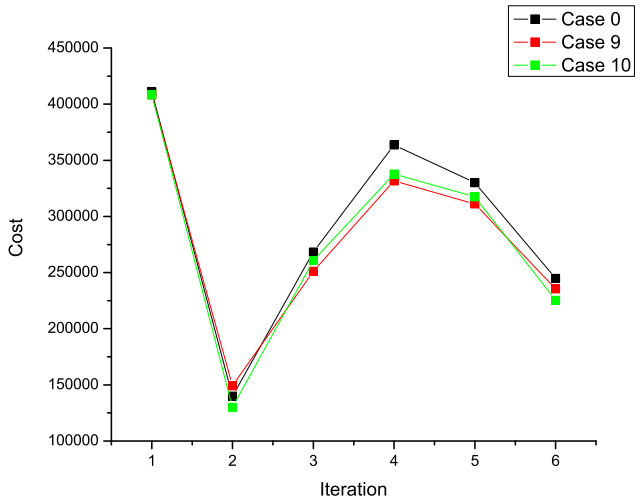
Instance 3	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6
Case 0	280,311.41	240,572.92	317,521.69	193,017.99	382,767.97	352,771.42
Case 9	284,562.78	232,554.51	315,407.93	206,842.45	351,449.54	334,900.26
Case 10	283,334.66	232,786.61	320,218.60	203,525.28	357,109.05	322,154.05

Figure 22: Cost on each day

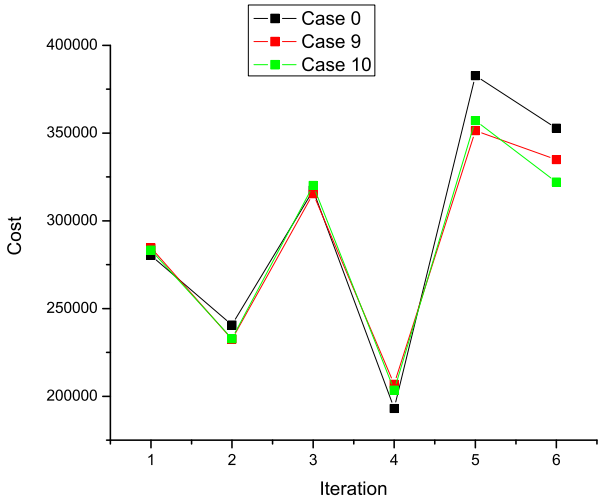
p-value of this observation is 0.03, which is less than the usual significance level 0.05. Therefore, we can reject the null hypothesis.



(a) Instance 1



(b) Instance 2



(c) Instance 3

Figure 23: Cost on each day

CHAPTER V

DATA ANALYSIS AND STOCHASTIC MAINTENANCE

Demand in our scheduling problem includes customer-requested flights and maintenance requests that are tied to specific airplanes. In this chapter, we will consider the analysis of demand data and model that deals with stochastic maintenance.

5.1 Analysis of Demand

When we solve an instance in practice, for example, an instance that spans two days, it is common that some customer-requested flights of the second day are not revealed yet. For example, they will be available at the end of the first day, at which time we need to re-solve the problem with newly added demand. With better knowledge of demand, we can generate better demand samples that are very desired if we model this 2-stage process with a stochastic model.

Moreover, better understanding of demand also helps our deterministic model in the previous chapter. If demand has patterns, then we can study the possibilities that these patterns may help to simplify the model or the implementation in practice.

Note that a demand has many attributes, such as departure airport, departure time, arrival airport, arrival time, duration etc. In the following, we will consider the set of the actual demands of one year from a major fractional airline company, which contains around 30,000 demands, and use R 2.9 for the analysis and graphics.

Figure 24 shows the correlations of attributes of a demand. The attributes in Figure 24 include the following: "Duration" is the duration of a demand in minutes; "OwnerID" denotes the customer who made the flight request; "InLon" and "InLat" are the longitude and latitude of the arrival airport; set of attribute for arrival time contains the followings: "InTimeM" is the arrival time in minutes; "InMonth",

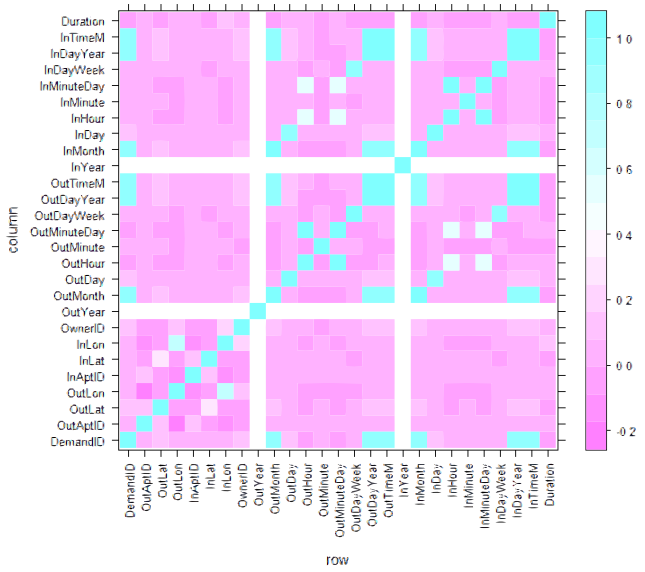


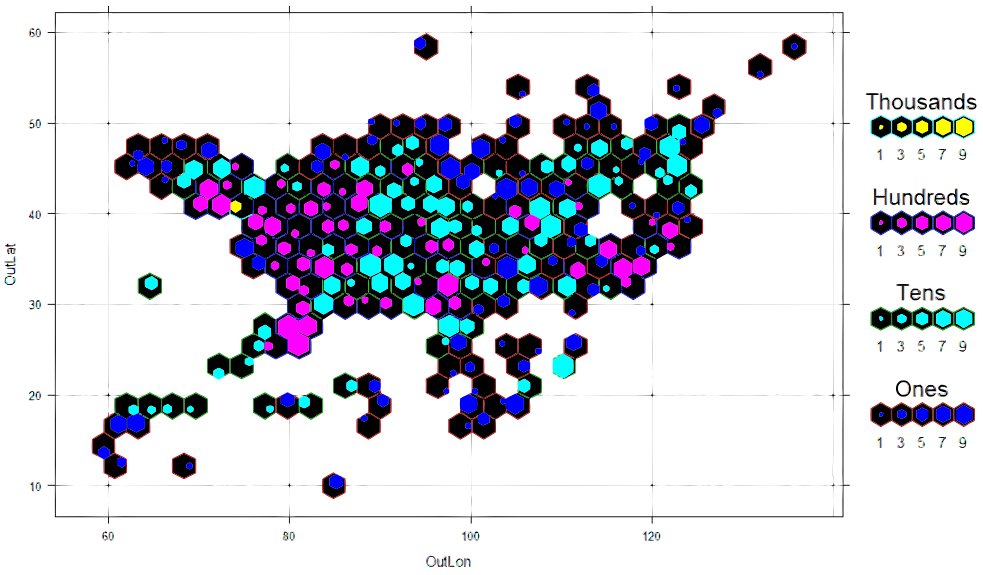
Figure 24: Correlations of demand attributes

"InDay", "InHour" and "InMinutes" describe the date and time of the arrival time; "InDayYear" and "InDayWeek" are the day of the year and day of the week of the arrival time; Similarly defined is the set of attributes for departure airport and departure time.

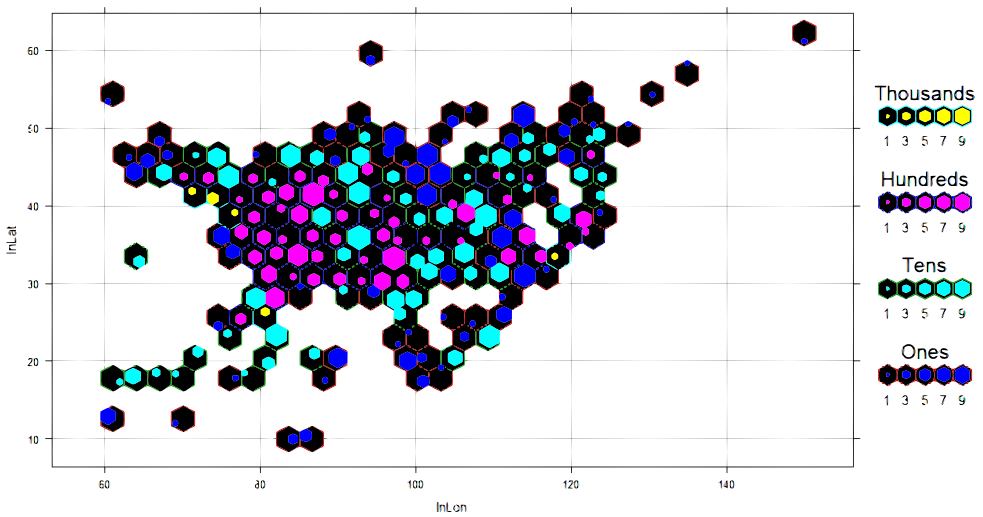
It follows from Figure 24 that the strong correlations are the trivial ones. For example, "OutMonth" and "InMonth".

When we consider airports, we want to know whether some airports are more heavily used than others. Figure 25 shows the scatter plot of airports in term of number of demands departing and arriving at an airport. The x-axis denotes longitude and y-axis denotes latitude, and the color in the plot denotes the density (number of demands). It follows from Figure 25 that the more heavily used airports, in the range of hundreds of demands, are still spread out.

Demands requested from the same owner are likely to have patterns, because an owner may have preferences. For example, an owner may have preferences for airport, departure time, departure day of the week, or duration of the requested flight.



(a) Departure Airports



(b) Arrival Airports

Figure 25: Airports of demands

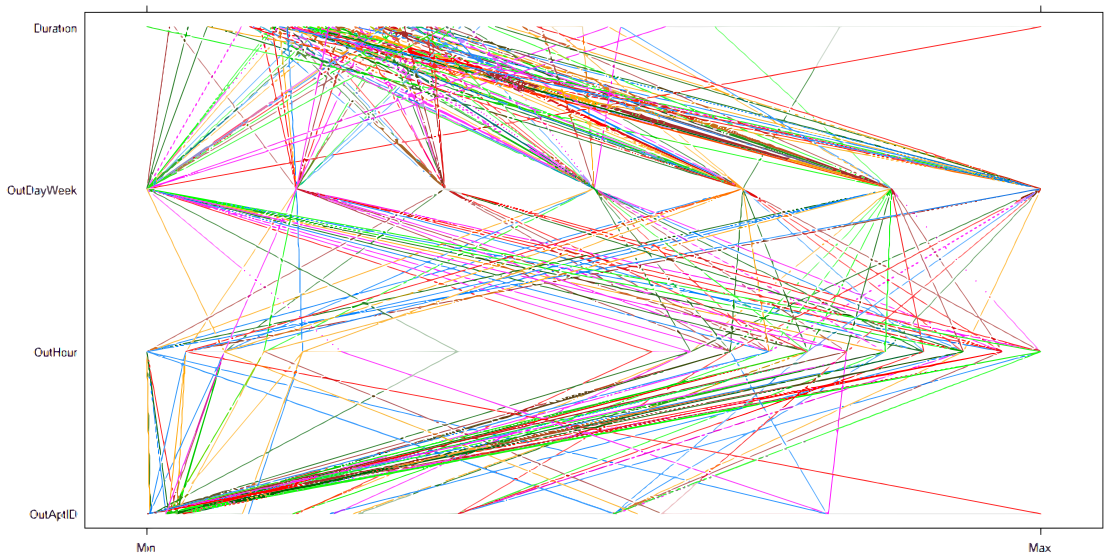


Figure 26: Demands of owner 1

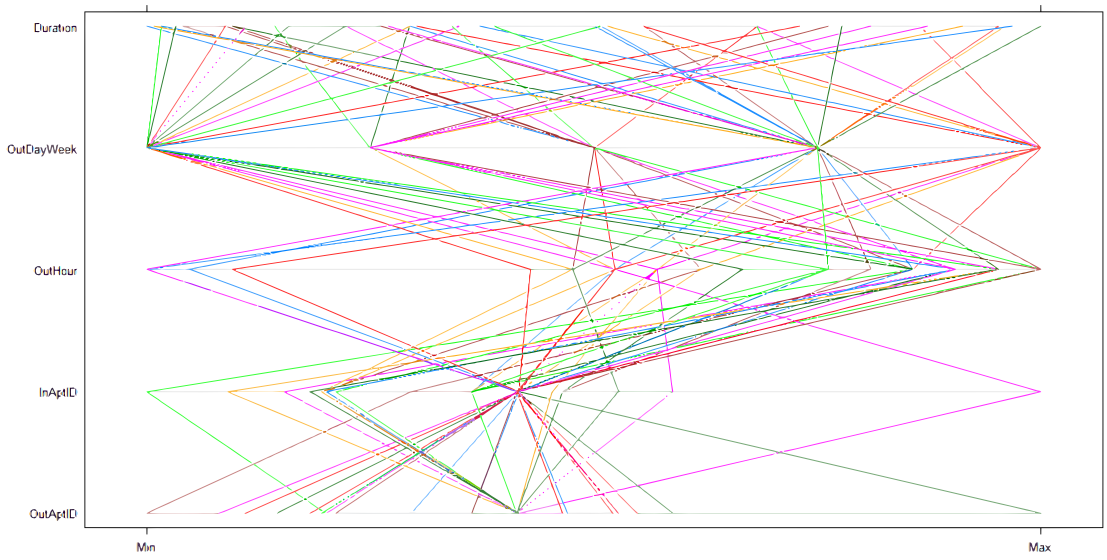


Figure 27: Demands of owner 2

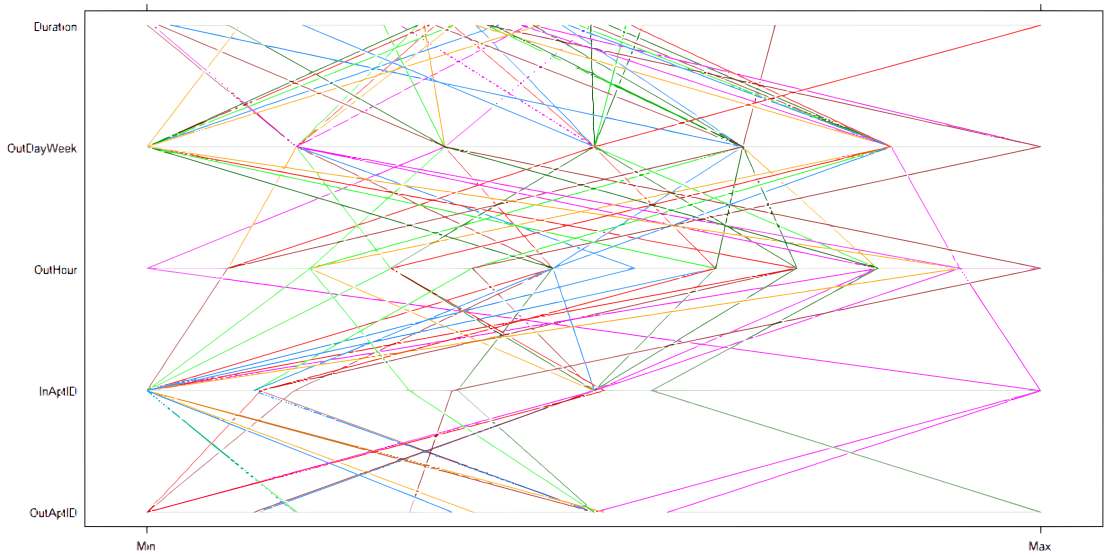


Figure 28: Demands of owner 3

We consider three owners who requested most flights during the year, and study if there are patterns and preferences in their flights. Consider Figure 26, and let us show how the plot is made. There are four x-axes in Figure 26: "Duration", "OutDayWeek", "OutHour" and "OutAptID", representing duration of the flight, day of the week of the departure time, hours (0-23) of the departure time and departure airport ID. Owner 1 has 174 demands during the year. The range of each axis is the range of the corresponding attribute over all 174 demands. For each demand, a line with unique color connects corresponding point on each axis in a top-down order.

From Figure 26, it follows that a few airports (near the bottom left) are heavily used than others. From Figure 27, the pattern for owner 2 is that there are many round trips. In Figure 28, many demands arrive at one of the three airports.

Density plot in Figure 29 shows that the distribution of demand durations is skewed with heavy tails, and the QQ-plot against normal distribution shows that the distribution is not likely to be normal. Moreover, for demands on each day of the week, the density plots in Figure 30 show that they are not significantly different from

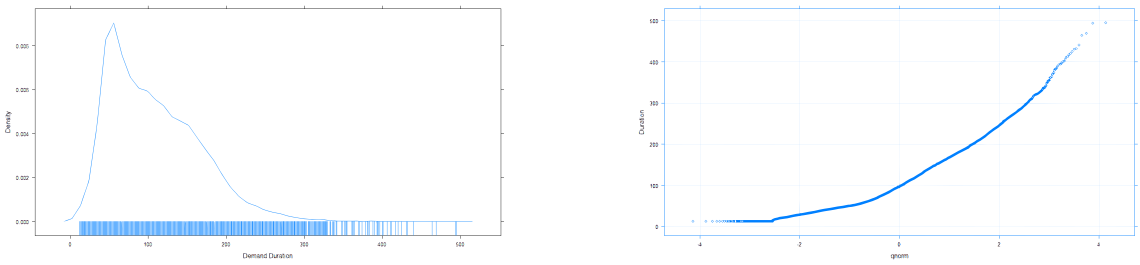


Figure 29: Demand duration

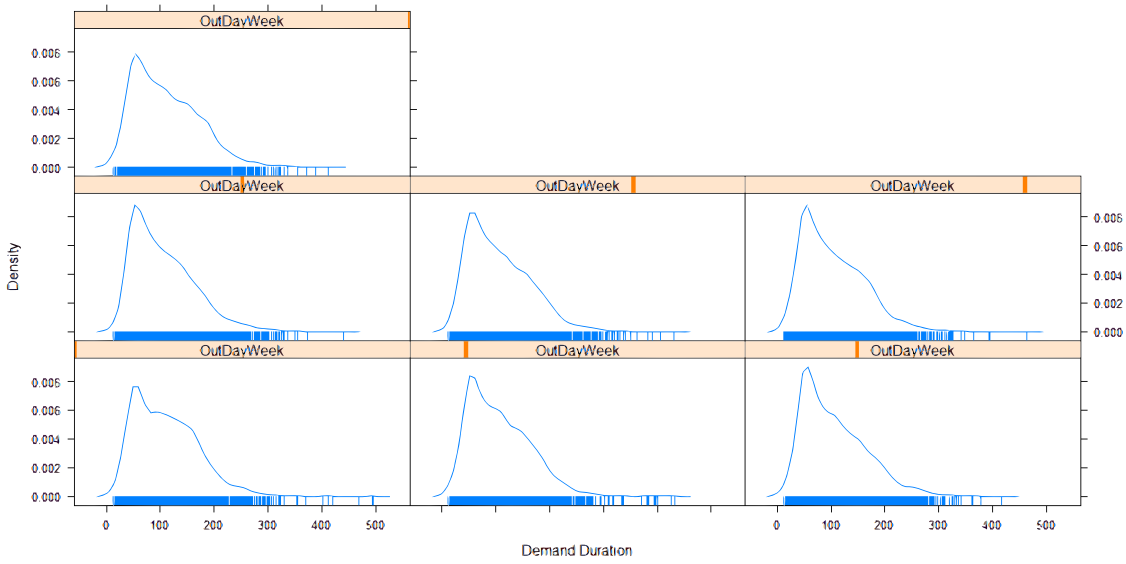


Figure 30: Demand duration of each day

each other.

Figure 31 demonstrates the departure time of demands. The x-axes are hours of a day (0-23) and minutes (0-59), and the y-axis denotes number of demands. From Figure 31, we can see the peak hours of a day, and that demands are almost always requested to depart at 0 and 30.

Figure 32 shows number of demands on each day of week and on each day of the month. Note that there are very few demands on Saturday.

Figure 33 is the Q-Q plot for number of demands of a day. It implies that the distribution for number of demands of a day is similar to a normal distribution.

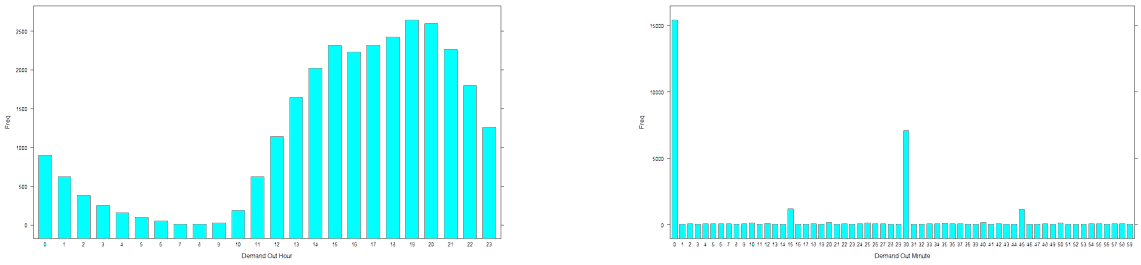


Figure 31: Demand departure time

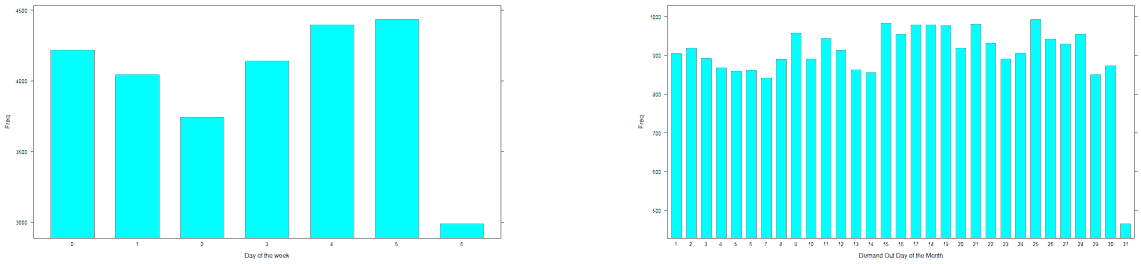


Figure 32: Number of demands

Number of demands on each day of the year is a time series as plotted in Figure 34. When considering the correlogram of this time series (Figure 35), we can see that numbers of demands have significant correlations in multiple of seven days, i.e. weeks. After fitting a few models, including the auto-ARIMA model supplied by the forecast package in R, we find that AR(21), the autoregessive model of order 21, is the best fit. This is verified by the correlogram of the residuals in Figure 35, which are very similar to white noises.

5.2 Stochastic Maintenance

In practice, the deterministic model in the previous chapter is often used in a rolling-horizon procedure. Given the current schedule, when there is new data (for example, newly added demand), or when there are changes to availabilities of the current resources (for example, unscheduled maintenance), we need to re-solve the model with updated information to generate a new schedule. In this section, we will consider the case of unscheduled maintenance, and propose more robust schedules by modifying

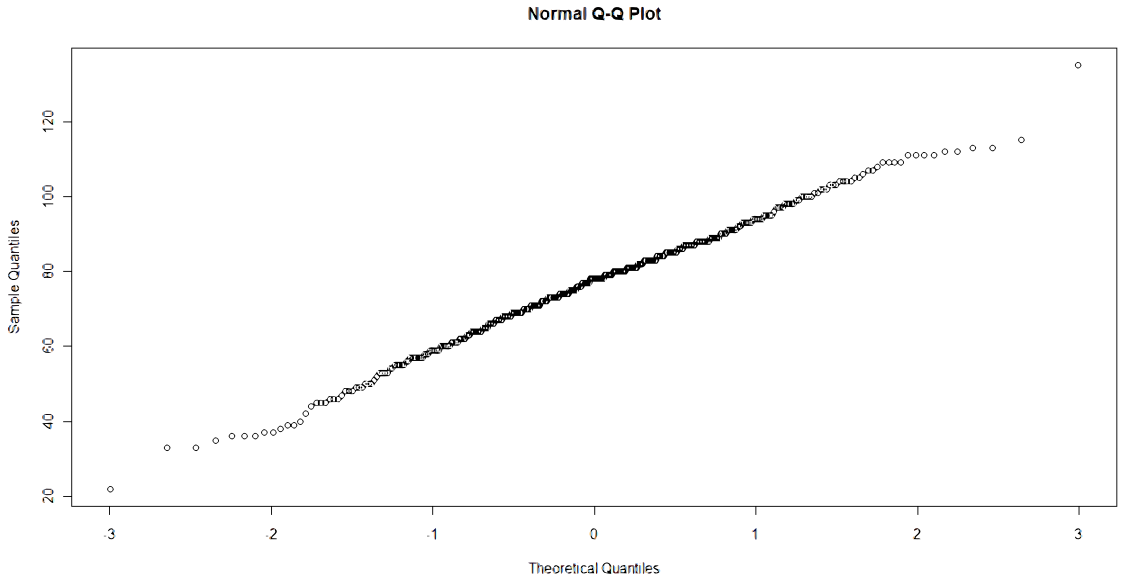


Figure 33: Compare distribution of number of demands with normal distribution

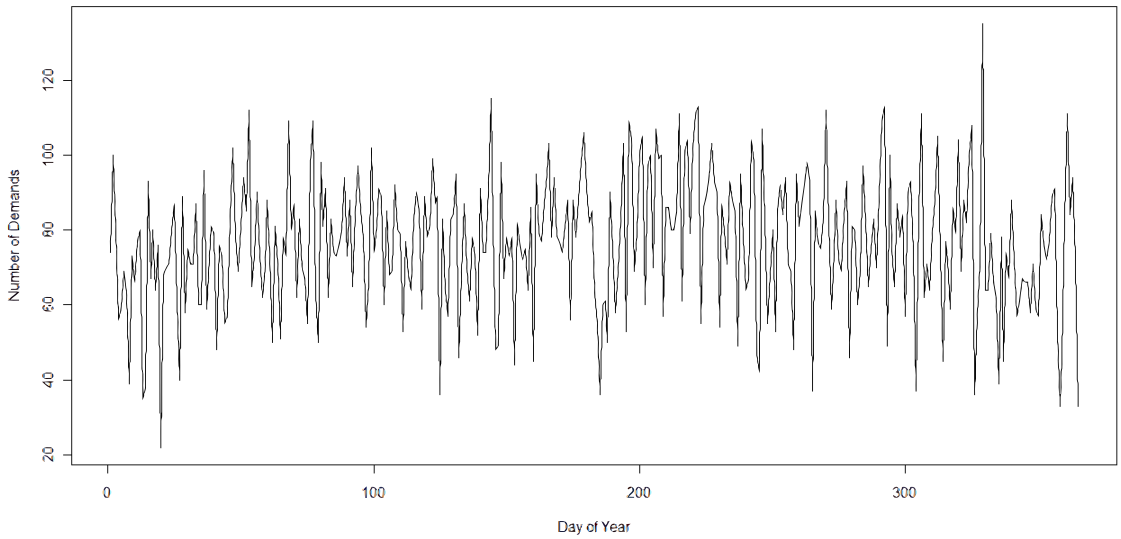


Figure 34: Number of demands of a year

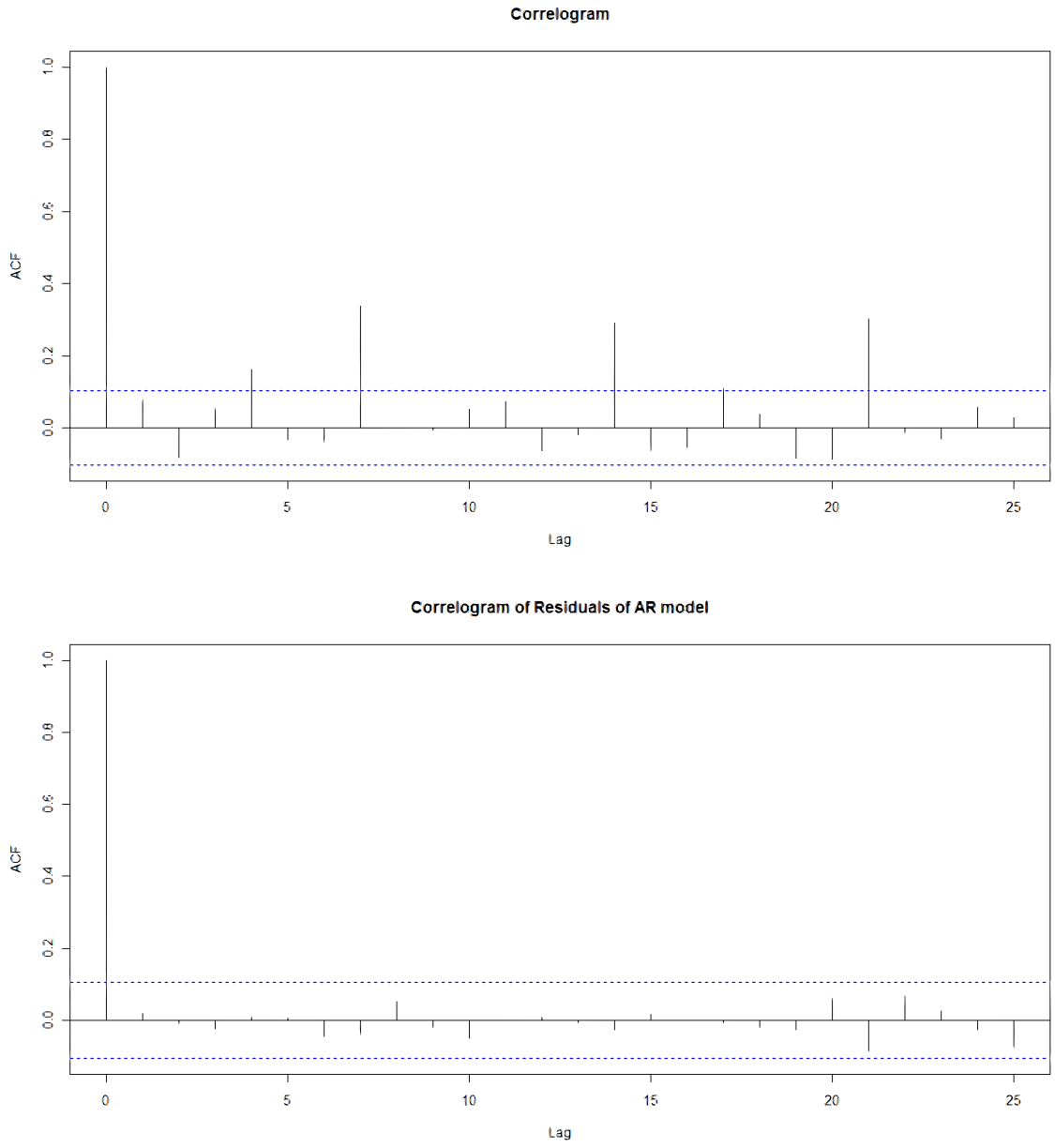


Figure 35: AR(21) model

the model in the previous chapter.

When an airplane a has an unscheduled maintenance at time t, it will be hold for maintenance checks and repairs, and will be released some time after t. If leg g is assigned to airplane a after time t in current schedule, then it is possible that, when airplane a is released, it is too late for airplane a to be able to fly leg g. So we need to generate a new schedule and try to recover leg g with another airplane. Total cost is likely to increase in this recovery process.

For easier presentation of ideas, when we assume that there is an unscheduled maintenance at time t, we assume the followings:

- at time t, we know that this maintenance is needed;
- this maintenance starts at t;
- its duration is known at time t and will not change.

In practice, we may know that an unscheduled maintenance is needed at time t, but the maintenance actually starts at time $t+\delta$, $\delta > 0$. Then after a maintenance starts, it may end sooner or later than expected. For example, it may depend on how soon the parts are obtained.

In order to reduce the negative impact of unscheduled maintenance, a natural and intuitive idea is to reserve some airplanes as spare/backup airplanes, which will not be assigned to any legs in the current schedule. The purpose of these airplanes is to recover the legs that are affected by unscheduled maintenance. The question then is how to choose these backup airplanes.

In practice, this decision is often done manually before solving the model. Therefore, marginal cost of reserving an airplane can only be estimated from the scheduler's experience. In this section, we will generalize this idea, and integrate decisions of reserving backup airplanes and crews into the model, so the model produce more robust schedules with respect to unscheduled maintenance.

5.2.1 Formulation

We will modify the MIP formulation (39) by adding *recovery columns* and *demand covering rows*.

Let $\mathbb{A}^r \subseteq \mathbb{A}$ be the set of backup airplane candidates. A backup airplane candidate can be either reserved as a backup airplane or assigned to fly legs, but not both.

Theoretically, after an airplane is available, it can be assigned to a leg l, and then after leg l, it is reserved as a backup airplane. However, in this case, we expect this airplane to be available at a certain airport in the future, i.e. we assumed that this airplane has no unscheduled maintenance after it is available and before leg l ends to interrupt leg l. This assumption may not hold in general, unless we have more knowledge about unscheduled maintenance.

We will choose backup airplane candidates that will not have unscheduled maintenance before it is reserved (details will be explained in the next section). Similarly, let $\mathbb{C}^r \subseteq \mathbb{C}$ be the set of backup pilots candidates.

Let $D^r \subset \mathbb{D}$ be the set of demands that we need to *cover* with backup airplanes and pilots. A demand $d \in D^r$ is *covered*, if there exist an airplane in $\mathbb{A}^r$ and a pair of pilots in $\mathbb{C}^r$ such that it is feasible for the pilots to pick up the airplane and reposition it to the departing airport of demand d before departure time of demand d. It means that this pair of pilots have enough available duty time and block time to travel to the airplane and fly the airplane to the departure airport of demand d if needed, and their fleet types is compatible with demand d. Note that we do not require that this pair of pilots has enough duty time and block time left to fly demand d, because if demand d needs to be recovered due to an unscheduled maintenance, then the pilots who are originally assigned to demand d are likely still available.

Let T^r be the set of all recovery columns. Each recovery column contains a pickup arc, which includes a pair of pilots in $\mathbb{C}^r$, an airplane in $\mathbb{A}^r$, and the information about how the pilots travel and rest to pick up the airplane. In addition, a recovery column

also contains demands in D^r that can be covered by the pickup arc in this recovery column. The cost of a recovery column includes the crew's travel cost and overtime cost, but does not include the repositioning cost to recover the demands or cost of the demands.

For each recovery column $t \in T^r$, binary variable α_t denotes whether column t is in the solution. For each demand $d \in D^r$, non-negative variable β_d denotes at what level demand d are covered by recovery columns in the solution.

We will add variables $\{\alpha_t : t \in T^r\}$ and $\{\beta_d : d \in D^r\}$ and a new type of constraints to MIP (39) in the previous chapter to get the following formulation:

$$Min \quad \sum_{t\in T} c_t x_t + \sum_{d\in \mathbb{D}} c_d y_d + \sum_{t\in T^r} c_t \alpha_t + \sum_{d\in D^r} p_d \beta_d \tag{60}$$

$$s.t. \quad \sum_{t\in T} m_{(a,t)} x_t + \sum_{t\in T^r} m^r_{(a,t)} \alpha_t \leq 1, \qquad \forall\, a \in \mathbb{A} \tag{61}$$

$$\sum_{t\in T} m_{(c,t)} x_t + \sum_{t\in T^r} m^r_{(c,t)} \alpha_t \leq 1, \qquad \forall\, c \in \mathbb{C} \tag{62}$$

$$\sum_{t\in T} m_{(d,t)} x_t + y_d = 1, \qquad \forall\, d \in \mathbb{D}$$

$$\sum_{t\in T} m^g_{(w,t)} x_t \leq 0, \qquad \forall\, f \in F, g \in A^f, w \in W^g$$

$$\sum_{t\in T} m_{(u,t)} x_t + \sum_{p,q\in U, p\neq q} m^u_{(p,q)} z_{(p,q)} \leq 0, \qquad \forall\, u \in U$$

$$\sum_{t\in T} m_{(p,q,t)} x_t + z_{(p,q)} \leq 0, \qquad \forall\, p, q \in U \text{ and } p \neq q$$

$$\sum_{t\in T^r} m_{(d,t)} \alpha_t + \beta_d \geq 1, \qquad \forall\, d \in D^r \tag{63}$$

$$x_t = \{0,1\}, \forall\, t \in T,\ y_d = \{0,1\}, \forall\, d \in D,$$

$$\alpha_t = \{0,1\}, \forall\, t \in T^r,\ \beta_d \geq 0, \forall\, d \in D^r, \tag{64}$$

$$z_{(p,q)} \geq 0, \forall\, p, q \in U \text{ and } p \neq q.$$

In the objective function (60), c_t for $t \in T^r$ is the cost of recovery column t, and p_d

for demand $d \in D^r$ is the penalty associated with covering demand d. In constraints (61), $m^r_{(a,t)} = 1$ if $a \in A^r$, $t \in T^r$ and recovery column t contains airplane a. Similarly, in constraints (62), $m^r_{(c,t)} = 1$ if $c \in C^r$, $t \in T^r$ and recovery column t contains pilot c.

Let us consider constraints (63). The purpose of constraints (63) is to cover each demand in D^r with recovery columns. A natural choice for the coefficient $m_{(d,t)}$ seems to be setting $m_{(d,t)} = 1$, if $d \in D^r$, $t \in T^r$ and demand d is contained in recovery column t. However, this choice of $m_{(d,t)}$ does not reflect levels of covering from recovery columns. For example, comparing to a demand in a recovery column that has 10 demands, a demand in a recovery column that has 30 demands should be considered as less covered.

Therefore, we define a covering parameter ξ that is a positive integer, and define coefficient $m_{(d,t)}$ in constraints (63) to be $\frac{\xi}{n(t)}$, where $n(t)$ is the number of demand in recovery column t.

Comparing to MIP (39), there are a new type of columns (recovery columns) and a new set of constraints (63) in MIP (60). Therefore, in order to solve MIP (39), we need to generate recovery columns in the column generation process, and we also need to consider duals corresponding to constraints (63). However, we can enumerate all recovery columns upfront and add them to the initial restricted master problem, and then use the same solution approach as in solving MIP (39). With our input data, there are typically a few thousands of recovery columns, so the explicit formulation with all recovery columns is possible and efficient.

5.2.2 Computational Experiments

In this section, we will show how to use a rolling horizon procedure to simulate unscheduled maintenance in practice, how to generate samples of unscheduled maintenance and how to determine rolling periods. And then we will present the computational results.

5.2.2.1 Use a rolling horizon procedure to simulate unscheduled maintenance in practice

We can use a rolling horizon procedure to simulate the handling of unscheduled maintenance in practice as follows: start with a feasible schedule, produced by either MIP (39) or MIP (60); if the next unscheduled maintenance takes place at time t, then assume that we are at time t now, and fix the partial schedule up to time t; re-solve either MIP (39) or MIP (60) with updated airplane availabilities and a new planning horizon that starts at time t.

Therefore, we need to re-solve the model whenever an unscheduled maintenance starts (we have assumed that we know about an unscheduled maintenance when it starts). In other words, let $M = \{m_0, m_1, \ldots, m_n\}$ be the set of all unscheduled maintenance requests, and let $T = \{t_0, t_1, \ldots, t_n\}$ be the set of times, where t_i is the start time of $m_i \in M$. Then T is the set of rolling periods, and we need to re-solve the model at each t_i, for $0 \leq i \leq n$.

5.2.2.2 Generate a sample of the unscheduled maintenance for the rolling horizon procedure

Our goal is to compare the effectiveness of two different models: MIP (39) and MIP (60), in the rolling horizon procedure with unscheduled maintenance. We will run two models with the same input data, and ideally with the same set of unscheduled maintenance requests.

In addition to start time t_i, an unscheduled maintenance m_i needs to specify an

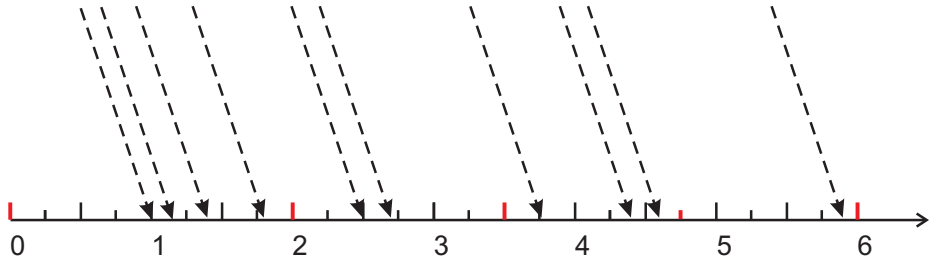


Figure 36: Rolling horizon intervals

airplane a_i or an airport ap_i. However, if an airplane a_i is specified, then it may not be feasible to perform a maintenance for airplane a_i at time t_i. This is because of the following: in the solving of either MIP (39) or MIP (60) in the previous horizon, we do not have any knowledge of the maintenance m_i. So in the output schedule of the previous horizon, airplane a_i may be flying or at another airport at time t_i, which makes the maintenance m_i infeasible. This also implies that we can not use the actual unscheduled maintenance for the rolling horizon procedure.

With the above considerations, we will assume that each $m_i \in M$ is associated with a demand d_i. It implies that an unscheduled maintenance will happen to whichever airplane assigned to d_i, maintenance start time t_i is the arrival time of d_i, and the airport a_i is the arrival airport of d_i.

With the above assumptions, we randomly choose such demands to in order to generate sample of the unscheduled maintenance. However, we need to know how many unscheduled maintenance requests to generate, i.e., how many demands to choose, and the duration of each unscheduled maintenance.

This is where we use the actual maintenance data, which is included in the input data. We will generate the same number of unscheduled maintenance requests in the sample. Moreover, we consider the duration of an unscheduled maintenance as a random variable, and fit a distribution with the actual maintenance data. Then for each unscheduled maintenance in the sample, we samples this distribution to get the duration.

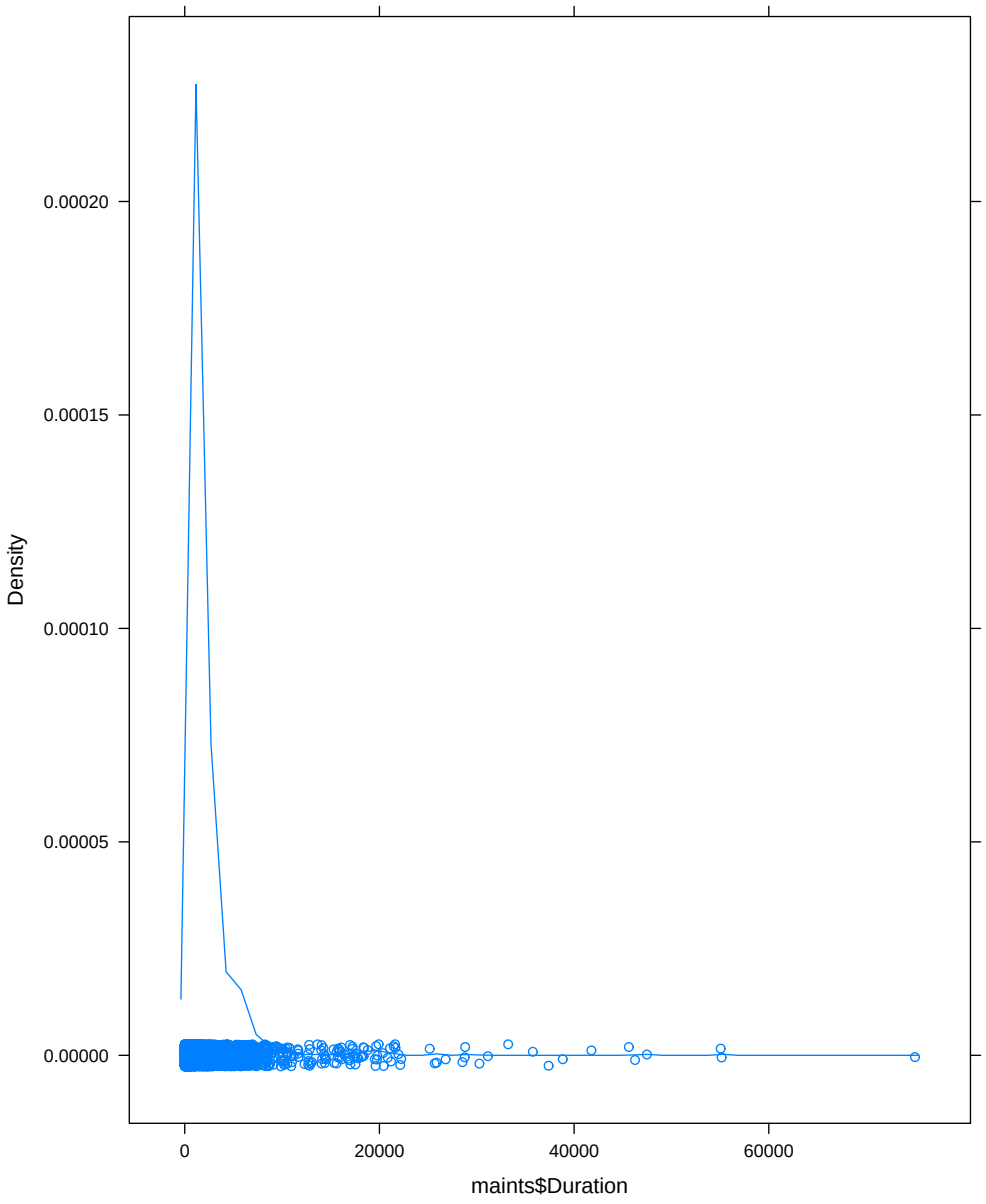


Figure 37: Density plot of maintenance durations

More specifically, we use the actual maintenance data that consists of all maintenance requests from one year, then use functions in R 2.9.1 to find a best fit Weibull distribution for the maintenance duration. Density plot for durations in the actual maintenance data is shown in Figure 37, and the best fit Weibull distribution has shape parameter equal to 0.69 and scale parameter equal to 883.1.

5.2.2.3 Determine the rolling periods

Under our assumption about the unscheduled maintenance, we can show that we do not need to re-solve the model at each t_i, i.e. we can have less number of rolling periods.

Assume that current rolling period and horizon start at time t, and let t_j be the next unscheduled maintenance after time t. If we re-solve the model at time $t_j + turnTime$, then all the unscheduled maintenance requests between t and $t_j + turnTime$ are taking place after t_j. Moreover, when we are at time $t_j + turnTime$, all the airplanes that had unscheduled maintenance in $[t_j, t_j + turnTime]$ are still on the ground. This is because that under our assumption, an unscheduled maintenance in $[t_j, t_j + turnTime]$ corresponds to a demand arriving at time $t_k \in [t_j, t_j + turnTime]$. Therefore, if there is a leg after this demand in current schedule, then this leg must depart after time $t_k + turnTime > t_j + turnTime$. Therefore, we can re-solve the model at time $t_j + turnTime$.

Figure 36 shows a small example about how to generate rolling periods.

5.2.2.4 Determine the parameters in MIP (60)

We need to choose the set of backup airplane candidates: A^r. Let t_i be start time of the current horizon. If an airplane a is available before time t_i, then any unscheduled maintenance happened to airplane a have been realized and finished before it is available. So airplane a is available on the ground at time t_i, and it will not have any unscheduled maintenance after time t_i and before a flight is assigned to it.

Moreover, if an airplane a is available after time t_i, then in the previous horizon, airplane a was assigned to a flight g that departs before time t_i and arrives after t_i. If flight g is a customer demand, then based on our assumption, it is possible that there will be an unscheduled maintenance associated with g, but we do not know about it at the current time t_i. If flight g is a repositioning leg, then airplane a will not have unscheduled maintenance if it stays on the ground after flight g.

In summary, we choose all airplanes that are available now, or available in the future and after a repositioning leg, to be the set A^r. Since we do not assume uncertainty for pilots, we set C^r to be the set of all available pilots. The set of to-be-covered demands D^r contains all demands that will depart within 12 hours from current time t_i.

Note that two parameters control the covering level of demands: penalty p_d in the objective function (60) and covering parameter ξ in constraints (63). When p_d is higher, or ξ is smaller, the covering level that the model requires is higher.

5.2.2.5 Computational results

Computations consist of two steps: we first test different values of the covering parameters, then we choose the value that has the best performance for a more extensive test.

In Table 3, columns except the last one correspond to instances. More specifically, each column corresponds to a 3-day instance from actual data and a sample of the unscheduled maintenance for this instance. The overall planning window is three days, and we consider the unscheduled maintenance during the first two days, i.e. the rolling horizon procedure stops at the end of the second day.

In the six instances contained in Table 3, number of pilots ranges from 274 to 301, number of airplanes ranges from 82 to 85, and number of demands, including maintenance, ranges from 283 to 337. These instances do not include peak days, and

	I1	I2	I3	I4	I5	I6	Average
$p_d = 50, \xi = \infty$	1.86%	-1.91%	1.12%	1.71%	4.85%	2.32%	1.66%
$p_d = 100, \xi = \infty$	6.26%	-5.06%	-2.97%	2.03%	0.05%	0.42%	0.12%
$p_d = 200, \xi = \infty$	5.08%	-1.78%	-0.93%	-4.57%	0.79%	0.21%	-0.20%
$p_d = 400, \xi = \infty$	8.15%	0.61%	-3.00%	-1.84%	0.50%	1.88%	1.05%
$p_d = 500, \xi = \infty$	7.84%	-1.98%	-2.64%	1.55%	-1.18%	0.51%	0.68%
$p_d = 700, \xi = \infty$	6.52%	-1.96%	-0.95%	-2.19%	0.33%	-0.37%	0.23%
$p_d = 50, \xi = 10$	4.80%	-4.31%	-1.27%	0.30%	3.01%	3.60%	1.02%
$p_d = 100, \xi = 10$	5.23%	-3.24%	-4.10%	0.08%	1.22%	3.95%	0.52%
$p_d = 200, \xi = 10$	5.10%	0.58%	1.57%	0.47%	1.33%	2.78%	1.97%
$p_d = 400, \xi = 10$	-1.67%	-3.57%	0.18%	0.77%	-0.76%	4.33%	-0.12%
$p_d = 500, \xi = 10$	2.40%	-1.18%	-3.14%	3.23%	1.74%	0.05%	0.52%
$p_d = 700, \xi = 10$	3.75%	-3.64%	-3.44%	0.43%	0.06%	0.72%	-0.35%

Table 3: Unscheduled Maintenance Parameters Test, 3-day instances

represent typical instances in the actual data. For each instance, the number of times we resolve the model, i.e. number of rolling periods is between 18 and 24.

Each row corresponds to a combination of the covering parameters ξ and p_d. $\xi = \infty$ means that the coefficient $m_{(d,t)} = \frac{\xi}{n(t)}$ in constraints (63) is set to be 1. Recovery columns typically contain less than 30 demands, so $\xi = \infty$ is roughly equivalent to setting $\xi = 30$. In other words, we tested two levels of the covering parameter ξ: 10 and 30. If $\xi = 10$, then a demand in a recovery column that contains 10 demands is considered as well covered and constraint (63) for this demand is satisfied.

We choose the penalty parameter p_d with range from 50 to 700. Note the cost of a recovery column only consists of the travelling costs of two pilots, and cost for a typical travel is less than 300 in our models. Therefore, when $\xi = \infty$, setting $p_d = 700$ means the following: if two pilots can travel to an airplane to form a recovery column that can cover at leat one demand, and if reserving them does not affect the cost of the other part of the schedule, then the model will choose them.

The total cost of a rolling horizon procedure is calculated as follows: at the start of a rolling period, we add the cost of the previous horizon (including charter cost)

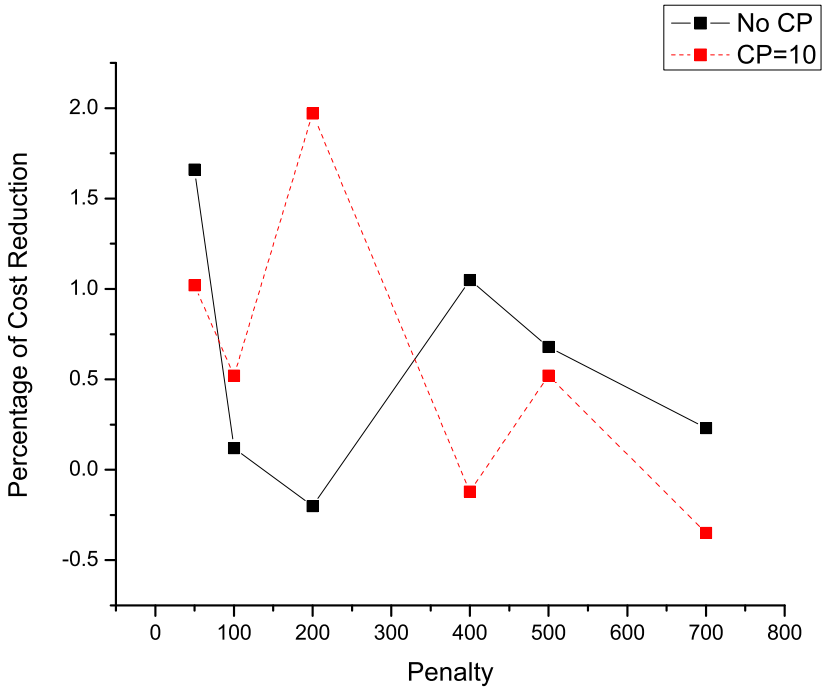


Figure 38: Effects of covering parameters

to the total cost. Therefore, this reflects the actual cost in the case of unscheduled maintenance in practice. Note that this also implies that the cost of the last rolling period is not included in the total cost. So total cost for a 3-day instance includes the total cost of the first two days, during which there is unscheduled maintenance, and the cost of the last day, which is the last rolling period, is not included.

Moreover, we have discussed how to determine rolling periods from the a given set of unscheduled maintenance requests. In order to reduce the impact of new demand and focus on the unscheduled maintenance, the end time of the horizon at the start of each rolling period is set to be the end of the following day. Therefore, we only see new demand at the end of a day.

Let us consider the upper left entry in Table 3. It means that we run both model (39) and model (60) in the rolling horizon procedure on instance 1 with $p_d = 50, \xi = \infty$, and value of this entry is the percentage of the total cost reduction from model

	1st Sample	2nd Sample	3rd Sample	Average
Instance 1	3.61%	2.60%	0.14%	2.12%
Instance 2	0.51%	2.46%	0.75%	1.24%
Instance 3	4.44%	0.44%	3.90%	2.93%
Instance 4	3.21%	1.63%	3.56%	2.80%

Table 4: Unscheduled Maintenance Test, 6-day instances

(39).

Figure 38 plotted the entries in the last column of Table 3, i.e., the average over all instances for each combination of parameters. The x-axis is p_d, and y-axis is the percentage of total cost reduction. Dotted line in Figure 38 corresponds to the case when $\xi = 10$, and the solid line corresponds to the case when $\xi = \infty$. It follows that the cost reduction is not increasing as p_d increases. However, note that the trends of these two lines are similar.

Another observation from Table 3 is that the case with $p_d = 200$ and $\xi = 10$ has the best cost reduction, and in this case, total cost reduces on each instance. We choose this combination and run both model (39) and model (60) on instances with longer overall planning window.

Table 4 shows that model MIP (60) with $p_d = 200$ and $\xi = 10$ works consistently better than model MIP (39), with the average cost reduction of 2.27 percent.

Rows in Table 4 correspond to instances, and columns, except for the last one, correspond to samples of the unscheduled maintenance. Note that the i-th column in Table 4 means that we generate the i-th sample for each instance, and it does not refer to a particular sample that is used for all instances. Each instance is a 6-day problem from the actual data, and they do not overlap. Number of pilots ranges from 276 to 314, number of airplanes ranges from 84 to 87, and total number of demands ranges from 429 to 464. These instances also do not include peak days, and for each instance, the number of times we resolve the model, i.e. number of rolling periods is between 43 and 52.

	Instance 1	Instance 2	Instance 3	Instance 4	Average
Repo Cost	1.04%	-4.60%	1.55%	-3.61%	-1.41%
Travel Cost	-2.58%	-1.04%	0.92%	-1.72%	-1.11%
Charter Cost	13.45%	12.47%	11.31%	23.21%	15.11%

Table 5: Costs Break Down, 6-day instances

Each entry in the first four columns of Table 5 reflects the reductions of repositioning, travel or charter cost, averaging over all three samples of the corresponding instance. An entry in the last column is the corresponding cost averaging over all instances. Table 5 shows that, with model (60), repositioning and travel cost slightly increase, while the charter cost decreases significantly, which suggests that more resources are available and utilized during the rolling horizon procedure, and shows the effectiveness of model (60) in dealing with unscheduled maintenance.

5.2.3 Future work

Although the computational results show that model (60) is effective in dealing with unscheduled maintenance, it also shows that the covering parameters have big impact on the solutions of model (60). One interesting question, also related to how to apply model (60) in practice, is how to determine best covering parameters, given the variety of instances in practice. With growing experience with model (60), schedulers should be able to choose appropriate covering parameters and see the advantage of model (60). Moreover, with the study of demand data and model with stochastic maintenance in this chapter, a stochastic programming approach may be used to deal with both new demand and unscheduled maintenance, and to better simulate the decision procedures in practice.

REFERENCES

[1] Barnhart, C., Belobaba, P., and Odoni, A. R., "Applications of operations research in the air transport industry," *Transportation Science*, vol. 37, pp. 368–391, NOV 2003.

[2] Barnhart, C., Cohn, A., Johnson, E., Klabjan, D., Nemhauser, G., and Vance, P., "Airline crew scheduling," in *HANDBOOK OF TRANSPORTATION SCIENCE* (Hall, R. W., ed.), pp. 517–560, KLUWER ACADEMIC PUBLISHERS, 2003.

[3] Barnhart, C. and Schneur, R., "Air network design for express shipment service," *Operations Research*, vol. 44, no. 6, pp. 852–863, 1996.

[4] Bertossi, A., Carraresl, P., and Gallo, G., "On some matching problems arising in vehicle scheduling models," *Networks*, vol. 17, pp. 271–281, 1987.

[5] Dantzig, G. B. and Wolfe, P., "Decomposition principle for linear programs," *Operations Research*, vol. 8, pp. 101–111, 1960.

[6] Desaulniers, G., Desrosiers, J., Dumas, Y., Solomon, M. M., and Soumis, F., "Daily aircraft routing and scheduling," *Management science*, vol. 43, no. 6, pp. 841–855, 1997.

[7] Desrosiers, J., Dumas, Y., Solomon, M. M., and Soumis, F., "Time constrained routing and scheduling," in *HANDBOOKS IN OR AND MS* (Ball, M., ed.), vol. 8, pp. 35–139, ELSEVIER SCIENCE B.V., 1995.

[8] Dumas, Y., Desrosiers, J., and Soumis, F., "The pickup and delivery problem with time windows," *European Journal of Operational Research*, no. 54, pp. 7–22, 1991.

[9] Erdmann, A., Nolte, A., Noltemeier, A., and Schrader, R., "Modeling and solving an airline schedule generation problem," *Annals of Operations Research*, no. 107, pp. 117–142, 2001.

[10] Espinoza, D., Garcia, R., Goycoolea, M., Nemhauser, G. L., and Savelsbergh, M. W. P., "Per-seat, on-demand air transportation part I: Problem description and an integer multicommodity flow model," *Transportation science*, vol. 42, no. 3, pp. 263–278, 2008.

[11] Espinoza, D., Garcia, R., Goycoolea, M., Nemhauser, G. L., and Savelsbergh, M. W. P., "Per-seat, on-demand air transportation part II: Parallel local search," *Transportation science*, vol. 42, no. 3, pp. 279–291, 2008.

[12] Garey, M. R. and Johnson, D. S., *Computers and intractability: A guide to the theory of NP-Completeness*. W. H. Freeman, 1979.

[13] Gilmore, P. C. and Gomory, R. E., "A linear programming approach to the cutting-stock problem," *Operations Research*, vol. 9, pp. 849–859, 1961.

[14] Gilmore, P. C. and Gomory, R. E., "A linear programming approach to the cutting-stock problem–part ii," *Operations Research*, vol. 11, pp. 863–888, 1963.

[15] Hicks, R., Madrid, R., Milligan, C., Pruneau, R., Kanaley, M., Dumas, Y., Lacroix, B., Desrosiers, J., and Soumis, F., "Bombardier flexjet significantly improves its fractional aircraft ownership operations," *Interfaces*, vol. 35, no. 1, pp. 49–60, 2005.

[16] Keskinocak, P. and Tayur, S., "Scheduling of time-shared jet aircraft," *Transportation Science*, vol. 32, no. 3, pp. 277–294, 1998.

[17] Kim, D. and Barnhart, C., "Flight schedule design for a charter airline," *Computers and Operations Research*, no. 34, pp. 1516–1531, 2007.

[18] Klabjan, D., "Large-scale models in the airline industry," in *COLUMN GENERATION* (DESAULNIERS, G., DESROSIERS, J., and SOLOMON, M. M., eds.), GERAD 25th Anniversary Series, pp. 163–195, Springer, 2005.

[19] Klabjan, D., Johnson, E., Nemhauser, G., Gelman, E., and Ramaswamy, S., "Airline crew scheduling with time windows and plane count constraints," *Transportation Science*, vol. 36, no. 3, pp. 337–348, 2002.

[20] Lubecke, M. E. and Desrosiers, J., "Selected topics in column generation," *Operations Research*, vol. 53, no. 6, pp. 1007–1023, 2005.

[21] Martin, C., Jones, D., and Keskinocak, P., "Optimizing on-demand aircraft schedules for fractional aircraft operators," *Interfaces*, vol. 33, no. 5, pp. 22–35, 2003.

[22] Mercier, A. and Soumis, F., "An integrated aircraft routing, crew scheduling and flight retiming model," *Computers and Operations Research*, vol. 34, pp. 2251–2265, 2007.

[23] Rexing, B., Barnhart, C., Kniker, T., Jarrah, A., and Krishnamurthy, N., "Airline fleet assignment with time windows," *Transportation Science*, no. 34, pp. 1–20, 2000.

[24] Ronen, D., "Scheduling charter aircraft," *Journal of the Operational Research Society*, vol. 51, pp. 258–262, 2000.

[25] Savelsbergh, M. W. P., "Local search in routing problems with time windows," *Annals of Operations Research*, vol. 4, pp. 285–305, 1985.

[26] Savelsbergh, M. W. P. and Sol, M., "The general pickup and delivery problem," *Transportation Science*, vol. 29, pp. 17–29, 1995.

[27] Yang, W., Karaesmen, I., Keskinocak, P., and Tayur, S., "Aircraft and crew scheduling for fractional ownership programs," *Annals of Operations Research*, vol. 159, pp. 415–431, Mar. 2008.

[28] Yao, Y., Ergun, O., Johnson, E., Schultz, W., and Singleton, J. M., "Strategic planning in fractional aircraft ownership programs," *European Journal of Operational Research*, vol. 189, pp. 526–539, 2008.